Hierarchical Continuous Time Dynamic Modelling for Psychology and the Social Sciences

Dissertation
zur Erlangung des akademischen Grades
Doctor rerum naturalium
(Dr. rer. nat.)
im Fach Psychologie

eingereicht an der Lebenswissenschaftlichen Fakultät der
Humboldt-Universität zu Berlin

von **Charles Conrad Driver**

Präsident der Humboldt-Universität zu Berlin:
Prof. Dr.-Ing. habil. Dr. Sabine Kunst

Dekan der Lebenswissenschaftlichen Fakultät:
Prof. Dr. rer. nat. Bernhard Grimm

Gutachter

1. Prof. Dr. Manuel C Voelkle
2. Prof. Dr. Ulman Lindenberger
3. Prof. Dr. Francis Tuerlinckx

Tag der mündlichen Prüfung: 12th October 2017

Acknowledgements

It's been a long and life changing road that sees me here in Berlin, far from the suburbs of my home town Brisbane, finalising this dissertation. Much gratitude to my advisor, Professor Manuel Voelkle, for many things. For giving me a start, one very wintry Berlin 'spring'. For countless hours of discussion after I would come banging on the door – discussion in which I slowly learned to formulate vague and broad intuitions about models, constructs, and change, into somewhat more concrete and understandable points. For coping with my sometimes 'rough and ready' approach, when more prudent mouths might have stayed silent. In general, for all the support, critique and argument when needed, and for allowing me much freedom to explore.

Thanks to: the Lifespan Psychology department, headed by Professor Ulman Lindenberger, within the Max Planck for Human Development, for the wonderful intellectual and financial support during this period; To my Formal Methods group leader, Andreas Brandmaier, and co-author Han Oud, for both being rich sources of insight; My colleagues and friends at the institute – in particular my office mate Janne Adolf, for many hours of interesting discussion and occasional combat, and Julian Karch, 'the guy who was actually trained in some of this technical stuff I never thought I would need when I first studied psychology'.

Some data used in this dissertation were made available by the German Socio-Economic Panel Study (SOEP) at the German Institute for Economic Research (DIW), Berlin. Software used in the data analyses and creation of this document includes knitr (Xie, 2014), R (R Core Team, 2014), RStudio (RStudio Team, 2016), Ωnyx (von Oertzen, Brandmaier & Tsang, 2015), OpenMx (Neale et al., 2016), Stan (Carpenter et al., 2017), and RStan (Stan Development Team, 2016a). A big thank you to all the open science, 'methodological terrorists', and 'replication crisis' folk in general – it's an inspiring time to work on methods and software!

I doubt I would be at this point without the input and inspiration of some others during my studies. A late but still valuable encounter with Nietzsche, along with some actual living students of the QUT Cliffhangers rock climbing club, made me think perhaps there were more interesting things to do in this world, and some study might help with that. Two professors during those studies in particular stand out, Bill von Hippel and Denny Borsboom. Bill communicated some of the fascinating things we can learn by observing the world, and Denny some of the fascinating difficulties that are involved in trying to do so. Two dear friends from Amsterdam with whom this passion for unravelling the difficulties developed, Anja Sommavilla and Janneke de Kort – though sadly Janneke is, and will be, missed...

To my brother and parents in Australia, thank you for the Skype love and airport collections, as my eyes readjust to real sunshine during Christmas escapes. And yes, Dad, thanks for playing all those board games with me when I was little, even if you did cheat. Thank you Mum for being different to Dad. Finally, although she didn't write that non-linear particle dynamics thesis I was hoping to benefit from, Giulia Paparo, for being my wonderful, expressive and supportive partner in this journey, and for helping convince me when it's time to sleep...

Abstract

Psychological states can be characterised as instantiations of ongoing and interrelated dynamic processes. By focusing on within subject changes, such an approach offers much promise for understanding, predicting, and influencing human cognition and behaviour. Given that changes take some time to come about, one might expect that the role of time would be prominent in such characterisations, yet commonly used statistical models for dynamics ignore this aspect. With this dissertation I endeavour to extend, and make practically applicable for psychology, the statistical approach of continuous time dynamic modelling, in which the role of time is made explicit. This approach couples latent processes governed by stochastic differential equations with a measurement model, and allows for the modelling of processes with uncertain dynamics, that are measured by noisy indicators, with variable time intervals between measurements. In this work I focus on *linear* stochastic differential equations, which, in the case of first order processes measured at consistent time intervals, are equivalent to the classic discrete time vector autoregressive model. A continuous time approach allows for much flexibility when gathering data, as time intervals between observations do not need to be consistent in order for the model to be appropriate – indeed, variability in time intervals may be helpful. Furthermore, differential equations allow for the parsimonious specification of complex dynamics within and between modelled processes. The structure of this dissertation is such that in Chapter 1, I discuss the nature of dynamic models, consider various approaches to handling multiple subjects, and detail a continuous time dynamic model with input effects (such as interventions) and a Gaussian measurement model. In Chapter 2, I describe the usage of the ctsem software for R developed as part of this dissertation, which provides a frequentist, mixed effects, structural equation modelling approach to estimation. Chapter 3 details a hierarchical Bayesian, fully random effects approach to estimation, allowing for subjects to differ not only in intercept parameters but in all characteristics of the measurement and dynamic models – while still benefiting from other subjects data for parameter estimation. Chapter 4 describes the usage of the Bayesian extension to the ctsem software. In Chapter 5 I consider the nature of experimental interventions in the continuous time dynamic modelling framework, and show approaches to address questions regarding the way interventions influence psychological processes over time, with questions such as 'how long does a treatment take to reach maximum effect', 'how does the shape of the effect change over time', and 'for whom is the effect strongest, or longest lasting'. Many examples using both frequentist and Bayesian forms of the ctsem software are given. For the final chapter I summarise the dissertation, consider limitations of the approaches offered, and provide some thoughts on possible future developments.

Keywords: *continuous time, dynamic model, stochastic differential equation, mixed-effects, panel data, state space model, hierarchical time series*

Zusammenfassung

Psychologische Zustände können als Instanzen andauernder und miteinander verknüpfter dynamischer Prozesse charakterisiert werden. Ein solcher, auf intraindividuelle Veränderungen fokussierender Ansatz erscheint vielversprechend für das Verständnis, die Vorhersage und die Beeinflussung menschlichen Erlebens und Verhaltens. Angesichts der Tatsache, dass das Zustandekommen von Veränderung Zeit benötigt, könnte man erwarten, dass die Rolle der Zeit in solchen Charakterisierungen prominent sei, doch häufig verwendete statistische Modelle für zeitliche Dynamiken vernachlässigen diesen Aspekt. Im Rahmen dieser Dissertation bemühe ich mich, den statistischen Ansatz der zeitkontinuierlichen dynamischen Modellierung, der die Rolle der Zeit explizit berücksichtigt, zu erweitern und praktisch anwendbar zu machen. Dieser Ansatz verbindet durch stochastische Differentialgleichungen getriebene latente Prozesse mit einem Messmodell und ermöglicht so die Modellierung von Prozessen mit ungewisser Dynamik, die durch fehlerbehaftete Indikatoren zu unregelmäßigen Zeitpunkten gemessen werden. In dieser Arbeit konzentriere ich mich auf *lineare* stochastische Differentialgleichungen, die, im Falle von in konsistenten Zeitintervallen gemessen Prozessen erster Ordnung, dem klassischen zeitdiskreten Vektorautoregressiven Modell entsprechen. Ein zeitkontinuierlicher Ansatz bietet hohe Flexibilität bei der Datenerhebung, da Zeitintervalle zwischen Beobachtungen nicht konsistent sein müssen, damit das Modell angemessen ist – tatsächlich kann Variabilität in Zeitintervallen sogar hilfreich sein. Darüber hinaus erlauben Differentialgleichungen die sparsame Spezifikation komplexer Dynamiken innerhalb und zwischen den modellierten Prozessen. Diese Dissertation ist so strukturiert, dass ich in Kapitel 1 die Natur dynamischer Modelle bespreche, verschiedene Ansätze zum Umgang mit mehreren Personen betrachte und ein zeitkontinuierliches dynamisches Modell mit Input-Effekten (wie Interventionen) und einem Gaußschen Messmodell detailliert darstelle. In Kapitel 2 beschreibe ich die Verwendung der Software ctsem für R, die als Teil dieser Dissertation entwickelt wurde und die Modellierung von Strukturgleichungen und Mixed-Effects über einen frequentistischen Schätzansatz realisiert. In Kapitel 3 stelle ich einen hierarchischen, komplett Random-Effects beinhaltenden Bayesschen Schätzansatz vor, unter dem sich Personen nicht nur in Interceptparametern, sondern in allen Charakteristika von Mess - und Prozessmodell unterscheiden können, wobei die Schätzung individueller Parameter trotzdem von den Daten aller Personen profitiert. Kapitel 4 beschreibt die Verwendung der Bayesschen Erweiterung der Software ctsem. In Kapitel 5 betrachte ich die Natur experimenteller Interventionen vor dem Hintergrund zeitkontinuierlicher dynamischer Modellierung und zeige Ansätze, die die Art und Weise adressieren, mit der Interventionen auf psychologische Prozesse über die Zeit wirken. Das berührt Fragen, wie: “Nach welcher Zeit zeigt eine Intervention ihre maximale Wirkung”, “Wie ändert sich die Form des Effektes im Laufe der Zeit” und “Für wen ist die Wirkung am stärksten oder dauert am längsten an”. Viele Beispiele, die sowohl frequentistische als auch bayessche Formen der Software ctsem verwenden, sind enthalten. Im letzten Kapitel fasse ich die Dissertation zusammen, zeige Limitationen der angebotenen Ansätze auf und stelle meine Gedanken zu möglichen zukünftigen Entwicklungen dar.

Contributions & Declaration of Independent Work

Portions of this dissertation have been prepared in collaboration with co-authors for publication. I am the primary contributor in all regards. For the most up to date versions, see the following:

Chapter 2, with various additions and changes, is available as a vignette in the R package ctsem, and is published as:

Driver, C. C., Oud, J. H. L., & Voelkle, M. C. (2017). Continuous time structural equation modeling with r package ctsem. *Journal of Statistical Software, 77(5)*. https://doi.org/10.18637/jss.v077.i05

Chapter 3, along with some content from Chapters 1 and 6, is currently in press:

Driver, C. C., & Voelkle, M. C. (In Press). Hierarchical Bayesian continuous time dynamic modeling. *Psychological Methods*.

Chapter 4 is also available as a vignette for the R package ctsem:

https://cran.r-project.org/web/packages/ctsem/vignettes/hierarchical.pdf

Except where otherwise stated, I, Charles Driver, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis. This doctoral thesis, or parts thereof, has not already been submitted as part of another, either accepted or rejected, doctoral thesis. I have not applied for a doctoral degree elsewhere and do not have a corresponding doctoral degree in this doctoral subject. The principles of the Humboldt University of Berlin have been complied with to ensure good scientific practice.

Contents

Chapter 1.

Introduction

Besides a few bumps and deviations in the road, psychologists have typically been interested in the internal cognitive and emotional states of people. Even though such states are not directly knowable, inference from observations to internal states has proved to be a useful approach to help understand people – whether one is a scientist, or one simply observes an angry look on a partners' face! But what is meant here by "help understand people"? In some situations, predicting some aspect of the future based only on *current observations* may be sufficient. In other situations, we may instead also wish to predict what would happen should we *intervene in some particular way*, or going yet further, we may want to *know how to intervene* to produce a specific result. All three questions, with increasing degrees of difficulty, can be addressed within a dynamic modelling framework. Such a framework considers the relation between fluctuating and interrelated *processes*. To make the situation more concrete, let us consider that angry looking partner. Attending to the coldness of voice and intensity of the angry glare might help to predict how angry they are, and perhaps how long the anger may last. Based on earlier observations, one might also be able to predict the change in anger were one to, for instance, shrug one's shoulders and go to the fridge for a drink. To achieve a quick reduction in anger however, one would likely require a sufficiently developed mental model of the initial and prolonging causes of the anger, such that one could act to reduce or eliminate that cause. Fundamental, though often left implicit, is the role of time – external events and internal psychological processes unfold over some time span, such that of course, anger now is more predictive of anger in a minute than anger in a week. In this dissertation I focus on extending and making practically applicable the statistical approach of hierarchical continuous time dynamic models, that do explicitly incorporate time, and also allow for modelling differences between individuals. With such models, the processes may be measured with a high level of uncertainty, by a range of different indicators, with any length of time between observations. The processes may change in ways that cannot be well predicted, and characteristics of the measurements and processes may be different in different people.

The statistical approaches and software developed in this dissertation are not only applicable to psychological phenomena – indeed similar models have been put to good and frequent use in engineering, with a classic example being the guidance of the Apollo 11 lunar module (Grewal & Andrews, 2010). However, the intention of the work is to usefully extend, and make practically applicable, the models for the sorts of use cases typically found in psychology and the social sciences. Some major features determining this include sparse data, noisy measurements, and dynamics which are at best understood very vaguely – all generally troublesome features for developing accurate models! Something of a saving grace is the ability to observe the processes of interest in multiple subjects, and leverage the similarity between subjects to improve model estimates. This latter feature of the data is however not straightforward to benefit from, because while subjects may be similar, the extent of similarity is uncertain, and must itself be estimated based on some model.

In the remainder of this chapter I discuss dynamic models and their use in psychology, consider approaches to estimating such models when data from multiple subjects is available,

and then specify and describe in detail the continuous time dynamic model, measurement model, and input model, used throughout this work. Building on this base, in Chapter 2, I describe the usage of the ctsem (Driver, Oud & Voelkle, 2017) software for R (R Core Team, 2014) developed as part of this dissertation, which provides a frequentist, mixed effects, structural equation modelling approach to estimation of the model. Chapter 3 details a hierarchical Bayesian, fully random effects approach to estimation, allowing for subjects to differ not only in intercept parameters but in all characteristics of the measurement and dynamic models, while still benefiting from other subjects data for estimation. Chapter 4 describes the usage of the Bayesian extension to the ctsem software. In Chapter 5 I consider the nature of experimental interventions in the continuous time framework, and show approaches to address questions such as 'how long does a treatment take to reach maximum effect', 'how does the shape of the effect change over time', and 'for whom is the effect strongest, or longest lasting'. In the final chapter I summarize the dissertation, considering limitations and avenues for further development, as well as strengths of the approach and areas where it may be fruitfully applied.

1.0.1. Dynamic models

Models containing dependencies on earlier states and stochastic inputs, such as the autoregressive and cross lagged model, offer a flexible and powerful approach to examine how psychological constructs function over time within a subject. For the sake of expediency I will refer to such models as *dynamic models*, but note that such a term could in general encompass a broader range of models than dealt with here. Similar to trajectory oriented approaches such as latent growth curves, dynamic models generally involve some smooth long term trajectory. Instead of treating deviations from a smooth overall trajectory as mere uninformative noise, however, such deviations can often be informative as to future states. So, while someone's life satisfaction last year doesn't completely determine their current life satisfaction, it does have some predictive value, over and above their typical level for the last 30 years. If we were to develop a formal model out of such a statement, we would have a simple dynamic model involving some dependence on past states, and a stable baseline state. Since such a model model cannot perfectly predict the way satisfaction may change in time, we will also need a variance term to quantify the degree of uncertainty regarding predictions – if satisfaction last year was substantially below baseline, we may expect that satisfaction is still at a level below baseline, but with some uncertainty. The more time that passes before we check satisfaction again, the greater the uncertainty, as more external events and internal processes that we cannot fully predict will occur. As such, the more time that passes, the less informative our previous observation will be, until at some point, the previous observation no longer helps our prediction.

Carrying on with the conceptual model development, at this point our model assumes we have a perfect measure of life satisfaction, but this is highly unlikely – regardless of exactly how we measure, the same internal satisfaction state could result in a range of measured outcomes due to spurious factors that are unrelated to satisfaction. When such measurement errors are left in the dynamic model, any parameter estimates regarding the dynamics of the true process we are interested in are likely to be biased. Consequently, *state-space* models (Hamilton, 1986) have been developed, which partition observed deviations from a prediction into innovation variance, representing meaningful but unpredictable fluctuations in the process itself, and measurement error, representing uncorrelated random noise in the observations. The innovation variance may be considered meaningful, in that although it represents unpredictable changes in the process, once such changes in the processes are observed, they are useful for future predictions of the states of the processes. This is in contrast to measurement error, which will not offer any predictive value for the future. By fitting such a dynamic model to the data, we can ask questions like 'how long does a change in this persons' life satisfaction typically persist?'. If we include a second set of observations, this time regarding their health, we can then ask 'to what

extent do changes in life satisfaction and health covary?', and also consider temporal dynamics between the two, with 'to what extent does a change in health predict a later change in life satisfaction, after controlling for shared sources of change?'. The major distinction between such an approach and a trajectory only model such as a latent growth curve, is that the relation between fluctuations is addressed, rather than only the shape of overall change. So while it may be that over 30 years, a person exhibits some general decline in health and apparently stable life satisfaction, when the relation between fluctuations is taken into account, one might instead see that life satisfaction tends to fluctuate downwards when health satisfaction does, then slowly recover due to other factors.

Common examples of dynamic models include the autoregressive and cross-lagged panel model, (Hertzog & Nesselroade, 2003; Finkel, 1995) autoregressive moving average models (Box, Jenkins, Reinsel & Ljung, 2015), or the latent change score formulations sometimes used (John J. McArdle, 2009), when they include innovation variance for each time point. They can be applied with observational or experimental data, or a mix of the two, and can be used to tell us to what extent changes in a psychological process are predictive of later changes of either the same process, or some other process of interest. Some examples of the usage of dynamic models within psychology include the analysis of symptom networks in psychopathology (Bringmann et al., 2013), differences in individuals‘ affective dynamics (Koval, Sütterlin & Kuppens, 2016), and the influence of partners on one another (Hamaker, Zhang & van der Maas, 2009).

1.0.2. Continuous time dynamic models

Within psychology and many fields, classical approaches to dynamic modeling have tended to rely on *discrete time* models, which generally rely on an assumption that the time intervals between measurements are all the same. A discrete time model directly estimates the relation (regression strength) between one measurement occasion and another, ignoring the time interval. If all measurement occasions have the same time interval between them, this can be fine. In many cases though the time intervals between occasions may differ, and if the same regression parameter is used to account for different time intervals, it is likely that the parameter is incorrect for some or all occasions – the relation between someone's earlier mood and later mood is likely to be quite different if we compare a ten minute interval to a three day interval, for instance. Obviously, parameter estimates and, thus, scientific conclusions, are biased when observation intervals vary and this is not adequately accounted for. In simple cases, so called phantom variables (Rindskopf, 1984), with missing values for all individuals could be added in order to artificially create equally spaced time intervals. For complex patterns of individually varying time intervals, however, this approach can become untenable (Voelkle & Oud, 2013).

Continuous time models overcome these problems, offering researchers the possibility to estimate parameters free from bias due to unequal intervals, easily compare between studies and datasets with different observation schedules, gather data with variable time intervals between observations, understand changes in observed effects over time, and parsimoniously specify complex dynamics.

Rather than estimate the regression between two measurement occasions directly, the continuous time approach instead estimates parameters that describe how the process changes at any particular moment, based on its value relative to a baseline. A mathematical function can then be used, in combination with the time interval, to determine the regression strength between two occasions with any time interval between them. This is a non-linear function, that operates in a way such that for first order (dependent only on the previous occasion, and not earlier occasions) discrete-time models with constant time intervals, the discrete and continuous time approaches are equivalent. The same of course holds true for other dynamic model parameters, such as the intercepts and innovation covariance matrix. The ability to naturally and exactly account for different time intervals means continuous time models are particularly

suited to the analysis of data from studies with different measurement schemes.

While accounting for different time intervals naturally is one obvious benefit, another reason for interest in continuous time models is that the parameters directly represent elements of differential equations. This is beneficial both in terms of allowing more meaningful interpretation of the parameters, as well as for more readily specifying higher order dynamics such as damped oscillations. More meaningful interpretation of continuous time dynamic models is possible because they are not just a model for the specific time points observed as with a discrete time model, but rather the model describes expected behavior of the processes at *any* time point, whether observed or not. Differential equation models describing more complex (e.g., higher than first order) dynamics have proven of some interest in psychology, with analysis of constructs such as working memory (Gasimova et al., 2014), emotion, (Chow, Ram, Boker, Fujita & Clore, 2005), and addiction (Grasman, Grasman & van der Maas, 2016). For some background on how differential equation models can be applied to psychological constructs, see Deboeck, Nicholson, Bergeman and Preacher (2013) and Boker (2001). Important to note here is that continuous time dynamic models are not the only form of dynamic model using differential equations, as approaches such as generalized local linear approximation (Boker, Deboeck, Edler & Keel, 2010) are also available. Rather, the terminology 'continuous time model' has typically been used to refer to dynamic models that explicitly incorporate the time interval in the equations, leading to an 'exact' solution, that does not require the specification of an embedding dimension in advance. For discussion on the distinction between 'approximate' and 'exact' approaches to estimation of differential equations, see Oud and Folmer (2011) and Steele and Ferrer (2011).

Even for models that explicitly incorporate the time interval, exact, analytic solutions to the stochastic differential equations are usually only available for *linear* stochastic differential equations involving a Gaussian noise term, which are the form of model we are concerned with here. For more complex forms of continuous time model, various approximations are typically necessary, as for instance in Särkkä, Hartikainen, Mbalawata and Haario (2013). For further discussion of the continuous time dynamic model in psychology, Oud (2002) and Voelkle, Oud, Davidov and Schmidt (2012) describes the basic model and applications to panel data, and Voelkle and Oud (2013) details higher order modelling applications such as the damped linear oscillator. A classic reference for stochastic differential equations more generally is the text from Gardiner (1985).

1.0.3. Hierarchical Dynamic models

While in some circumstances, a well developed dynamic model for a single subject may be important, in many situations scientists are instead interested in generalising from observed subjects to some population, as well as understanding differences between subjects. For such purposes, one typically needs repeated measures data from multiple subjects. How then, should data from different individuals be combined? Were all subjects exactly alike, combining the information would be very simple and effective. Were the subjects entirely distinct, with no similarities, combining the information would tell us nothing – an estimate of the average weight of a type of leaf, is not improved by including a rock in the sample. The estimation approaches herein endeavour to tread the middle ground, in that while differences between subjects are acknowledged, similarities are leveraged to improve estimation. Such an approach can broadly be called hierarchical dynamic modelling, a term encompassing both frequentist and Bayesian approaches. To understand how hierarchical dynamic models function, and the benefits they offer, it is helpful to first consider two extremes of possible approaches to dynamic models with multiple subjects. At one end of the continuum is panel models with a single set of fixed-effects parameters governing the dynamics of every subject – 'all leaves weigh exactly the same' – and at the other lies person-specific time series – 'one leaf is to another, as to a rock'.

Panel models containing autoregressive and cross-lagged parameters are regularly estimated with a single set of fixed-effects parameters governing the behavior of the dynamic system for many subjects – one assumes that the system characterizing the psychological processes of one subject is exactly the same as for other subjects[1]. Benefits to such an assumption are that the number of parameters in the model is relatively low, and the data from every individual is relevant for estimating every parameter. This assumption usually makes fitting the model to data much simpler, and can increase the precision and accuracy of parameter estimates when the assumption is valid. However, it is a very strong assumption and can result in highly biased parameter estimates. This has long been recognized (i.e., Balestra & Nerlove, 1966) to be the case in regards to the intercept parameter of the dynamic model, in that when subject specific differences in the average level of the processes are not accounted for, the temporal dynamics parameters (auto and cross effects) can be terribly biased – instead of representing some average of the temporal dynamics for all subjects, they instead become a mixture of the within-subject temporal dynamics and the between-subjects differences in average levels. While many poor implementations and claims of large cross effects and causality can still be found, this issue is at least widely recognized and readily resolved. See Halaby (2004) and Hamaker, Kuiper and Grasman (2015) for more details, and Molenaar (2004), Cattell (1963) for more on between and within subject issues in general.

At the other end of the spectrum is idiographic, or individual specific, time series approaches, where one fits the model separately for each subject (see for example Steele, Ferrer & Nesselroade, 2014). Such approaches ensure that the estimated parameters are directly applicable to a specific individual. With 'sufficiently large' amounts of data for each subject, this is likely to be the simplest and best approach. However, in applications of dynamic models there may be many correlated parameters, coupled with noisy measurements that are correlated in time, ensuring that a 'sufficiently large' amount of data per subject may in fact be 'unattainably large', particularly when one wishes to distinguish measurement error and relate multiple processes. Models fit with less than ideal amounts of data tend to suffer from finite sample biases, as for example in the autoregressive parameter when the number of time points is low (Marriott & Pope, 1954, 3/4), and also from higher uncertainty and more inaccurate point estimates. Were parameters independent of one another then inaccurate estimates of one parameter may be tolerated, but in dynamic models there are typically very strong dependencies between parameters. This means that when, as typically occurs, the parameter governing the subjects' base level is overfit and explains more observed variance than it should, the related auto-effect parameter is typically underestimated, in turn affecting many other parameters in the model. While some may be inclined to view the complete independence between models for each subject as a strength of the single-subject time series approach, there is little value to such independence if it also comes at the cost of making the best model for the subjects empirically unidentifiable, or the estimates substantially biased. Recent work by Liu (2017) demonstrates some of these issues. They compare a multilevel and person-specific approach to modelling multiple-subjects autoregressive time series, without measurement error. They examine the effect of time series length, number of subjects, distribution of coefficients, and model heterogeneity (differing order models). So long as the model converges and is not under specified for any subjects (i.e., contains lags of a high enough order), they find that the multilevel approach provides better estimates, even when distributional assumptions regarding the parameters are not met. Of course, not every possibility has been tested, and it is likely possible to find specific cases where this does not hold.

Hierarchical approaches to dynamic models, such as those from Lodewyckx, Tuerlinckx,

[1]Note that this is distinct from what has become known in econometrics as the 'fixed-effects panel model', which estimates unique intercept terms for every subject, with all other parameters constant across subjects (Halaby, 2004).

Kuppens, Allen and Sheeber (2011), are essentially a generalization which encompasses the two extremes already discussed – a single model for all individuals, or a distinct model for each. Such a hierarchical formulation could take shape either in a frequentist random-effects setting, or a hierarchical Bayesian setting.

Using a hierarchical formulation, instead of estimating either a single set of parameters or multiple sets of independent parameters, one can estimate *population distributions* for model parameters. It is common in hierarchical frequentist approaches to only estimate population distributions, while with Bayesian approaches it is more typical to simultaneously sample subject level parameters *from* the population distribution, with the population distribution essentially serving as a prior distribution for the subject level parameters. The latter approach can help to provide the intuition for the relation between person-specific, hierarchical, and single fixed-effect parameter set approaches. Using a hierarchical model, if one were to fix the variance of the population distribution in advance, one could see that as the variance approaches zero, the subject level parameter estimates get closer and closer to the fixed-effects model with the same set of parameter values for every subject. Conversely, as one fixed the population variance further and further towards infinity, subject level estimates would get closer and closer to those of the person-specific approach, in which the parameter estimates for one subject are not influenced at all by estimates of the others. The benefit of a hierarchical approach however is that rather than fixing the parameters of the population distribution in advance, the population distribution mean and variance can be estimated at the same time as subject level parameters – the extent of similarity across subjects is estimated, rather than fixed in advance to 'not at all similar' or 'completely the same'. An intuitive interpretation of this is that information from all other subjects serves as prior information to help parameter estimation for each specific subject. As is standard with Bayesian methods, one still specifies some prior distributions in advance, but in hierarchical Bayes models these are priors regarding our expectations for the *population distribution*, otherwise known as hyperpriors. So, some initial, possibly very vague, hyperpriors are specified regarding possible population distributions. These hyperpriors, coupled with data and a Bayesian inference procedure, result in an estimated population distribution for the parameters of each subjects' dynamic model. This estimated population distribution, coupled with the likelihood of the dynamic model parameters for a specific subject (calculated just as we would in the single-subject approach), gives the posterior distribution for the subjects' specific parameters.

In the frequentist setting, while it is relatively straightforward to include random-effects for intercept parameters, random-effects for regression and variance parameters of latent factors have typically been more problematic, due to the complex integration required (see for instance Delattre, Genon-Catalot & Samson, 2013; Leander, Almquist, Ahlström, Gabrielsson & Jirstrand, 2015). This *mixed-effects* approach is helpful in that stable differences in level between subjects – likely the largest source of bias due to unobserved heterogeneity – are accounted for, while maintaining a model that can be fit reasonably quickly using the well and commonly understood frequentist inference architecture. However, while to the best of our knowledge there are not the same concerns of large biases, if individual differences in temporal dynamics or variance parameters are not modelled it is of course impossible to ask questions regarding such individual differences, which are a key interest in many cases. Bayesian approaches offer good scope for random-effects over all parameters, and indeed hierarchical Bayesian discrete time dynamic models are implemented and in use, see for instance Schuurman (2016).

Once a hierarchical dynamic model is estimated, one may be interested in questions regarding the population distribution, one or multiple specific subjects, or one or multiple specific time points of a subject. Questions regarding the population distribution could relate to: means and variances of parameter distributions, such as "what is the average cross effect of health on life satisfaction", and "how much variance is there in the effect across people?" Also possible are questions regarding correlations between population distributions of different parameters,

or between parameters and covariates, such as, "do those who typically have worse health show a stronger cross-effect?" Instead of whole population questions, one could instead ask about, for example, the cross effect of health on life satisfaction of a specific subject. Or yet more specific, we might for instance want to predict the health of a subject at some particular time point in the future, given a particular life event and set of circumstances.

1.1. Continuous time dynamic model specification

Before discussing the approaches for handling multiple subjects in more detail, we must first describe the process dynamics for a single subject. These are governed by the stochastic differential equation:

$$\mathrm{d}\boldsymbol{\eta}(t) = \Big(\mathrm{A}\boldsymbol{\eta}(t) + \mathrm{b} + \mathrm{M}\boldsymbol{\chi}(t)\Big)\mathrm{d}t + \mathrm{G}\mathrm{d}\mathrm{W}(t) \tag{1.1}$$

Vector $\boldsymbol{\eta}(t) \in \mathbb{R}^{\upsilon}$ represents the state of the latent processes at time t, so $\mathrm{d}\boldsymbol{\eta}(t)$ simply means the direction of change, or gradient, of the latent processes at time t. The matrix $\mathrm{A} \in \mathbb{R}^{\upsilon\times\upsilon}$ is often referred to as the drift matrix, with auto effects on the diagonal and cross effects on the off-diagonals characterizing the temporal dynamics of the processes. Negative values on the auto effects are typical, and imply that as the latent state becomes more positive, a stronger negative influence on the expected change in the process occurs – the process tends to revert to a baseline (at least in the absence of other influences). Positive auto-effects imply an explosive process, in which deviations from a baseline do not dissipate, but rather multiply. Cross-effects between processes may be positive or negative, with a positive cross-effect in the first row and second column implying that as the second process becomes more positive, the direction of change in the first process also becomes more positive.

The expected auto and cross regression matrix for a given interval of time (i.e., a discrete time effect) can be calculated as per Equation 1.2. Figure 1.1 shows this calculation for a specific bivariate process over a range of time intervals.

$$\mathrm{A}^{*}_{\Delta t_u} = \mathrm{e}^{\mathrm{A}(t_u - t_{u-1})} \tag{1.2}$$

The * notation is used in this work to indicate a term that is the discrete time equivalent of the original continuous time parameter, for the time interval Δt_u. The time interval Δt_u is the time at measurement occasion u minus the time at occasion $u-1$, with U the set of measurement occasions u from 1 to the total number of measurement occasions, and $u = 1$ treated as occurring at time $t = 0$. $\mathrm{A}^{*}_{\Delta t_u}$ then contains the appropriate auto and cross regressions for the effect of latent processes $\boldsymbol{\eta}$ at measurement occasion $u - 1$ on $\boldsymbol{\eta}$ at measurement occasion u. So, for a simple univariate process with $\mathrm{A} = -.4$, a time scale of weeks, and a time interval of two weeks between measurement occasions, the discrete time autoregression would be $\mathrm{e}^{-.4(2)} = 0.45$. Note that here the exponential of the A matrix is equivalent to the regular univariate exponential, but once the A matrix has non-zero off-diagonals, this is no longer the case and a matrix exponential function is necessary.

In Equation 1.1 the continuous time intercept vector $\mathrm{b} \in \mathbb{R}^{\upsilon}$, provides a constant fixed input to the latent processes $\boldsymbol{\eta}$. In combination with temporal dynamics matrix A, this vector determines the long-term level at which the processes fluctuate around. Without the continuous time intercept the processes, if mean reverting, would simply fluctuate around zero. The discrete time intercept may be calculated as per Equation 1.3, though for interpretation purposes we are inclined to think the asymptotic level of the processes, $\mathrm{b}_{\Delta t_\infty}$, is more useful. The asymptotic process level is the level they tend towards over time, and the level they start at if the process is stationary.

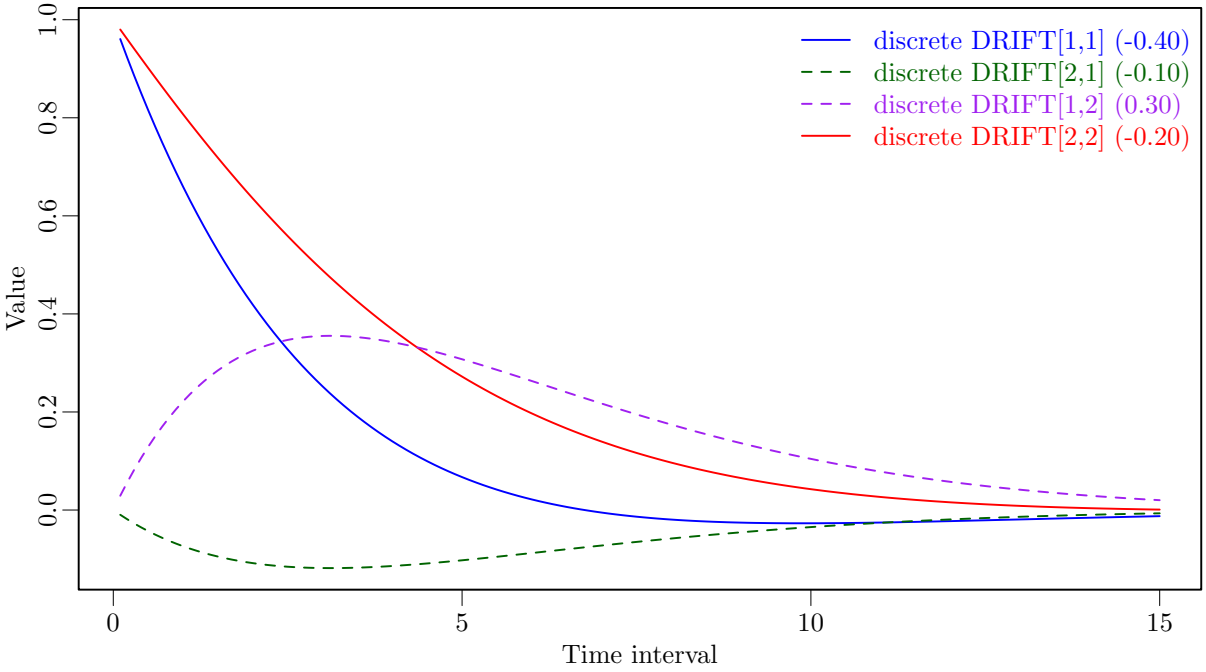


Figure 1.1.: Values of the *discrete DRIFT*, or auto and cross regression matrix, change depending on a function of the continuous time DRIFT matrix and the time interval between observations. The continuous time DRIFT values are shown in brackets in the legend.

$$\mathrm{b}^{*}_{\Delta t_u} = \mathrm{A}^{-1}(\mathrm{A}^{*}_{\Delta t_u} - \mathrm{I})\mathrm{b} \tag{1.3}$$

$$\mathrm{b}_{\Delta t_\infty} = -\mathrm{A}^{-1}\mathrm{b} \tag{1.4}$$

In Equation 1.1 the time dependent predictors $\boldsymbol{\chi}(t)$ represent exogenous inputs to the system, such as an intervention, that may vary over time and are independent of fluctuations in the system. Equation 1.1 shows a generalized form for time dependent predictors, that could be treated a variety of ways depending on the predictors assumed shape, or time course. We use a simple impulse form shown in Equation 1.5, in which the predictors are treated as impacting the processes only at the instant of an observation occasion u, and the effects then transmit through the system in accordance with A as usual. Such a form has the virtue that many alternative shapes are made possible via augmentation of the system state matrices – Chapter 5 discusses these aspects in depth.

$$\boldsymbol{\chi}(t) = \sum_{u \in \mathrm{U}} \mathrm{x}_u \delta(t - t_u) \tag{1.5}$$

Here, time dependent predictors $\mathrm{x}_u \in \mathbb{R}^l$ are observed at measurement occasions $u \in \mathrm{U}$. The Dirac delta function $\delta(t - t_u)$ is a generalized function that is ∞ at 0 and 0 elsewhere, yet has an integral of 1 (when 0 is in the range of integration). It is useful to model an impulse to a system, and here is scaled by the vector of time dependent predictors x_u. The effect of these impulses on processes $\boldsymbol{\eta}(t)$ is then $\mathrm{M} \in \mathbb{R}^{v \times l}$. Put simply, the equation means that at the measurement occasion u a time dependent predictor (e.g., intervention) is observed, the system processes spike upwards or downwards by Mx_u. For a typical intervention that probably only occurs once during the observation window, x_u would then be zero for every observation u except when the intervention occurred, where it could take on a dummy coding value such as

one, or could reflect the strength of the intervention. Since M is conceptualized as the effect of instantaneous impulses x, which only occur at occasions U and are not continuously present as for the processes $\boldsymbol{\eta}$, the discrete and continuous time forms are equivalent. This means that the effect of some intervention at measurement occasion $u = 3$ is simply $\mathrm{Mx}_{u=3}$ at the instant of the intervention, and at later measurement occasion $u = 4$, the remaining effect is $\mathrm{A}^*_{\Delta t_{u=4}}\mathrm{Mx}_{u=3}$. If the time interval between occasions 3 and 4 is $\Delta t = 2.30$, using Equation 1.2 this translates to $\mathrm{e}^{\mathrm{A}(2.30)}\mathrm{Mx}_{u=3}$.

In Equation 1.1, $\mathrm{W}(t) \in \mathbb{R}^{v}$ represents independent Wiener processes, with a Wiener process being a random-walk in continuous time. $\mathrm{dW}(t)$ is meaningful in the context of stochastic differential equations, and represents the stochastic noise term, an infinitesimally small increment of the Wiener process. Lower triangular matrix $\mathrm{G} \in \mathbb{R}^{v \times v}$ represents the effect of this noise on the change in $\boldsymbol{\eta}(t)$. Q, where $\mathrm{Q} = \mathrm{GG}^{\top}$, represents the variance-covariance matrix of this diffusion process in continuous time. Intuitively, one may think of $\mathrm{dW}(t)$ as random fluctuations of a fixed scale, and G as the effect of these fluctuations on the processes. To determine the innovation covariance matrix for a certain time interval, Equations 1.6 and 1.7 may be used. Figure 1.2 shows this calculation for a specific bivariate process over a range of time intervals.

$$\mathrm{Q}^*_{\Delta t_u} = \mathrm{Q}_{\Delta t_\infty} - \mathrm{A}^*_{\Delta t_u}\mathrm{Q}_{\Delta t_\infty}(\mathrm{A}^*_{\Delta t_u})^{\top} \tag{1.6}$$

$$\mathrm{Q}_{\Delta t_\infty} = \mathrm{irow}\big(-(\mathrm{A} \otimes \mathrm{I} + \mathrm{I} \otimes \mathrm{A})^{-1}\,\mathrm{row}(\mathrm{Q})\big) \tag{1.7}$$

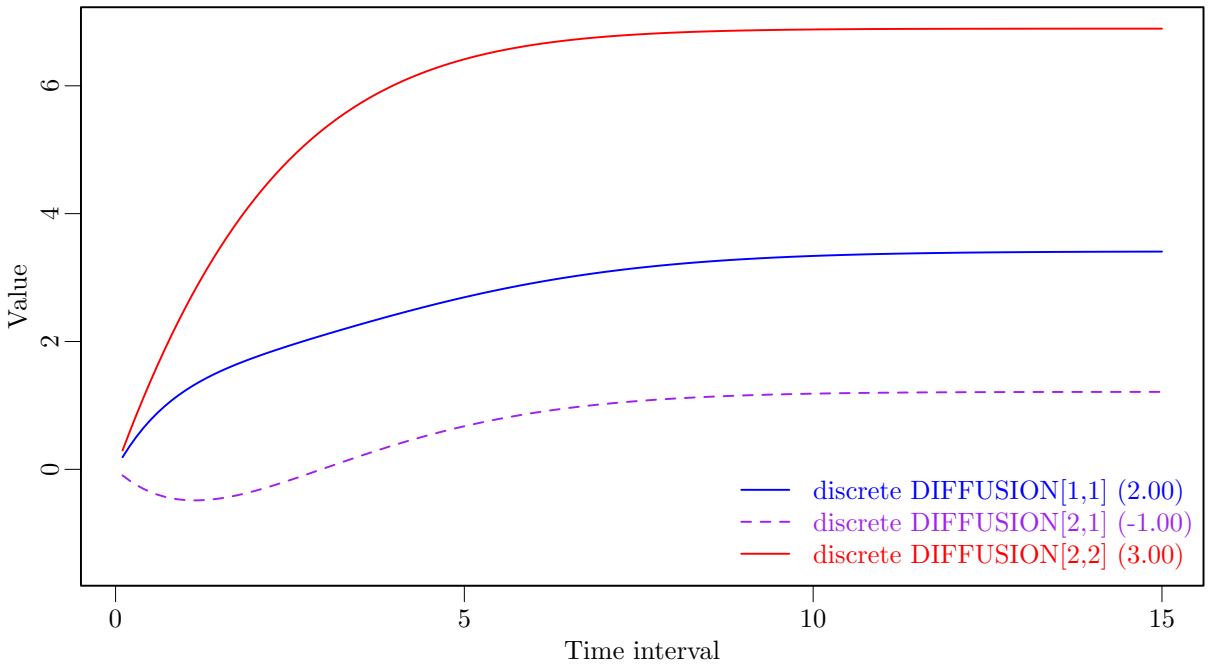


Figure 1.2.: Values of the innovation covariance matrix change depending on a function of the continuous time DRIFT and DIFFUSION matrices and the time interval between observations. The continuous time DIFFUSION values are shown in brackets in the legend (with DRIFT the same as for Figure 1.1). The values at higher time intervals are no longer changing substantially, thus representing the total within-person covariance matrix (or very close to it).

Equation 1.6 calculates the discrete time innovation covariance matrix for a given time interval. Intuitively, it shows that the discrete time innovation covariance matrix, which may be thought of as representing the amount and correlation of random noise added to the processes

over a specified time interval, equals the asymptotic, or total, innovation covariance matrix $Q_{\Delta t_\infty}$, minus the amount 'remaining' from the earlier measurement occasion. This remaining amount is determined by the temporal dynamics of A. $Q_{\Delta t_\infty}$ represents the innovation covariance matrix as Δt approaches infinity, which for a stationary process also represents the total within-subject variance covariance at any point in time. This asymptotic within-person covariance could also be thought of as the uncertainty about process states when no measurements of the processes exist. This matrix (as used in Tómasson, 2013) provides a computationally efficient basis for calculating the additional covariance $Q^*_{\Delta t_u}$ added to the system over the time interval Δt_u, as it only needs to be computed once. $\otimes$ denotes the Kronecker-product, row is an operation that takes elements of a matrix row wise and puts them in a column vector, and irow is the inverse of the row operation.

As the various discrete time calculations formula shown in this section rely on the matrix exponential seen in Equation 1.2, they can be difficult to intuitively understand. Plotting them with specific parameter values and increasing time interval Δt, can be a valuable tool, as this demonstrates how the implied discrete-time coefficients change as a function of the continuous time parameters and the time interval. An R script to perform such plots for a bivariate latent process model is shown in Appendix D.

1.1.1. Latent dynamic model — discrete time solution

To derive expectations for discretely sampled data, the stochastic differential Equation 1.1 may be solved and translated to a discrete time representation, for any observation $u \in \mathrm{U}$. Most components for this solution were already shown in Equations 1.2, 1.3, 1.5, 1.6, and 1.7, here we simply bring them together.

$$\boldsymbol{\eta}_u = \mathrm{A}^*_{\Delta t_u}\boldsymbol{\eta}_{u-1} + \mathrm{b}^*_{\Delta t_u} + \mathrm{M}\mathrm{x}_u + \boldsymbol{\zeta}_u \quad \boldsymbol{\zeta}_u \sim \mathrm{N}(0_v, \mathrm{Q}^*_{\Delta t_u}) \tag{1.8}$$

To reiterate, the * notation is used in this work to indicate a term that is the discrete time equivalent of the original continuous time parameter, for the time interval Δt_u. $\boldsymbol{\zeta}_u$ is the zero mean random error term for the processes at occasion u, which is distributed according to a multivariate normal with covariance $\mathrm{Q}^*_{\Delta t_u}$. The recursive nature of this solution means that at the first measurement occasion $u = 1$, the system must be initialized in some way, with $\mathrm{A}^*_{\Delta t_u}\boldsymbol{\eta}_{u-1}$ replaced by $\boldsymbol{\eta}_{t0}$, and $\mathrm{Q}^*_{\Delta t_u}$ replaced by Q^*_{t0}. These initial states and covariances are later referred to as T0MEANS and T0VAR respectively.

1.1.2. Measurement model

While in principle, non-Gaussian generalizations are possible, for the purposes of this work the latent process vector $\boldsymbol{\eta}(t)$ has the linear measurement model:

$$\mathrm{y}(t) = \Lambda\boldsymbol{\eta}(t) + \boldsymbol{\tau} + \boldsymbol{\epsilon}(t) \quad \text{where } \boldsymbol{\epsilon}(t) \sim \mathrm{N}(0_c, \Theta) \tag{1.9}$$

$\mathrm{y}(t) \in \mathbb{R}^c$ is the c length vector of manifest variables, $\Lambda \in \mathbb{R}^{c\times v}$ represents the factor loadings, and $\boldsymbol{\tau} \in \mathbb{R}^c$ the manifest intercepts. The manifest residual vector $\boldsymbol{\epsilon} \in \mathbb{R}^c$ has covariance matrix $\Theta \in \mathbb{R}^{c\times c}$. When such a measurement model is used to adequately account for non-zero equilibrium levels in the data, the continuous time intercept b may be unnecessary – accounting for such via the measurement model has the virtue that the measurement parameters are not dependent on the temporal dynamics, making optimization or sampling easier.

Chapter 2.

Continuous Time Structural Equation Modelling With R Package ctsem

In this chapter we introduce ctsem (Driver et al., 2017), an R package for continuous time structural equation modelling of panel ($N > 1$) and time series ($N = 1$) data, using full information maximum likelihood. By interfacing to OpenMx, ctsem combines the flexible specification of structural equation models with the enhanced data gathering opportunities and improved estimation of continuous time models. ctsem can estimate relationships over time for multiple latent processes, measured by multiple noisy indicators with varying time intervals between observations. Within and between effects are estimated simultaneously by modelling both observed covariates and unobserved heterogeneity in the intercepts. Exogenous shocks with different shapes, group differences, higher order diffusion effects and oscillating processes can all be simply modelled.

While there are already a range of packages that deal with continuous time (stochastic differential equation) models in R, most focus on single subject applications. These include sde (Iacus, 2015), yuima (Brouste et al., 2014), SIM.DiffProc (Boukhetala & Guidoum, 2014), cts (Z. Wang, 2013), POMP (King et al., 2010). For multi-subject approaches, OpenMx (Neale et al., 2015) now includes the function `mxExpectationStateSpaceContinuousTime`, which can be combined with the function `mxFitFunctionMultigroup` for fixed effects based group analysis. ctsem is focused on providing an accessible workflow, for full information maximum likelihood estimation of continuous time multivariate autoregressive models with random intercepts, and is useable for both time series and panel data. Using ctsem, one may specify: Cross lagged panel models; latent growth curve models; random intercepts at the latent or manifest level; damped oscillators; dynamic factor analysis models; constant or time dependent exogenous predictors; continuous time ARMAX models from the time series tradition; multiple groups or individuals with different parameters; or any combination of the preceding. First order models should be generally equivalent to discrete time first order models, if there is no variability in time intervals. For an example of this equivalence in regards to dual change score models see Voelkle and Oud (2015). For an R script and plot comparing estimates of a simple autoregressive model using ctsem and other packages, see Appendix A.

This chapter is organised as follows: in Section 2.1, we describe the continuous time structural equation models dealt with in the ctsem package. In Section 2.2 we will show how to install ctsem and give an overview of the package. In Section 2.3, we will review different data structures and discuss the role of time in continuous time models. In Section 2.4, we will show how to specify continuous time models in ctsem, followed by a discussion of model estimation and testing in Section 2.5. In Section 2.6 we will discuss various extensions of basic continuous time models, including unobserved heterogeneity, time dependent and time independent exogenous predictors, time series, multiple group models, higher order, and oscillating models. We will finish with some discussion of various specification options and tips for model fitting in Section 2.7.

2.1. Continuous time structural equation models: fundamentals

The class of continuous time structural equation models implemented in ctsem is represented by the multivariate stochastic differential equation:

$$\mathrm{d}\boldsymbol{\eta}_i(t) = \Big(\mathrm{A}\boldsymbol{\eta}_i(t) + \mathrm{b}_i + \mathrm{M}\boldsymbol{\chi}_i(t)\Big)\mathrm{d}t + \mathrm{G}\mathrm{d}\mathrm{W}_i(t) \quad \text{where } \mathrm{b}_i(t) \sim \mathrm{N}(\boldsymbol{\kappa}, \Upsilon) \tag{2.1}$$

With measurement model:

$$\mathrm{y}_i(t) = \Lambda\boldsymbol{\eta}_i(t) + \boldsymbol{\tau}_i + \boldsymbol{\epsilon}_i(t) \quad \text{where } \boldsymbol{\epsilon}_i(t) \sim \mathrm{N}(0, \Theta), \quad \text{and } \boldsymbol{\tau}_i \sim \mathrm{N}(\Gamma, \Psi) \tag{2.2}$$

These equations are very similar to Equation 1.1 and Equation 1.9, which are comprehensively explained in Chapter 1, beginning on page 7. The difference is we now explicitly include the subject indicator i, denoting which elements may vary across subjects in the SEM approach, and the relevant distributions.

The long term level of processes $\boldsymbol{\eta}_i(t)$ is determined by the v-length vector of random variables b_i, with $\mathrm{b}_i \sim \mathrm{N}(\boldsymbol{\kappa}, \Upsilon)$. Vector $\boldsymbol{\kappa} \in \mathbb{R}^v$ denotes the expectation for the continuous time intercepts, and matrix $\Upsilon \in \mathbb{R}^{v \times v}$ the covariance across subjects. Possible usage and interpretation of such heterogeneity is described in the initial portion of Section 2.6.1), but in general can be thought of as representing stable individual characteristics, that cause the processes to fluctuate around different levels. For this chapter, we will often refer to such as *traits*.

The c length vector Γ is the expected value of the subject specific manifest intercepts $\boldsymbol{\tau}_i$, which are distributed across subjects according to covariance matrix $\Psi \in \mathbb{R}^{c \times c}$. Possible usage and interpretation of such heterogeneity is described in the latter portion of Section 2.6.1), but in general can be thought of as representing stable individual characteristics that cause measurements to be higher or lower, given the same latent process state. For this chapter, we will often refer to such as *manifest traits*.

2.1.1. Continuous time and SEM

Continuous time models have already been implemented as structural equation models, using either non-linear algebraic constraints (Oud & Jansen, 2000) or linear approximations of the matrix exponential (Oud, 2002). Our formulation uses either the SEM RAM (reticular action model) specification as per J. Jack McArdle and McDonald (1984), or the state space form recently added to OpenMx (Neale et al., 2015; Hunter, 2014). For details on the equivalence and differences between SEM and state space modelling techniques, see Chow, Ho, Hamaker and Dolan (2010). ctsem translates user specified input matrices and switches into an OpenMx model consisting of continuous time parameter matrices, algebraic transformations of these matrices to aid optimization (See Section 2.5), and algebraic transforms from the continuous time parameters to discrete time parameters for every unique time interval. Expected means and covariance matrices are then generated for each individual according to the specified inputs, constraints, and observed timing data. Optimization using either the Kalman filter or row-wise full information maximum likelihood (FIML) within OpenMx is used to estimate the parameters, typically with a first pass using a penalty term (or prior) to find a region of high probability without extreme parameter values, and then a second FIML pass (i.e., without penalties) using the first as starting values.

To see exactly how the various matrices are transformed into a RAM SEM, one may run the following code after ctsem is installed (See Section 2.2). This example comprises two latent processes, three observed indicators, a time dependent predictor, and two time independent predictors, across three time points of observation.

```
data("datastructure")
datastructure
semModel<-ctModel(n.latent=2, n.manifest=3, TRAITVAR="auto",
  n.TIpred=2, n.TDpred=1, Tpoints=3,
  LAMBDA=matrix(c(1,"lambda21", 0, 0,1,0),nrow=3))
semFit<-ctFit(datastructure, semModel, nofit=TRUE)
semFit$mxobj$A$labels
semFit$mxobj$S$labels
semFit$mxobj$M$labels
semFit$mxobj$F$values
```

For more detailed information on the specification of continuous time structural equation models, the reader is referred to Oud and Jansen (2000), Arnold (1974), Singer (1998), Voelkle et al. (2012). Note that while earlier incarnations of continuous time modelling focused on approaches to implement the matrix exponential, OpenMx now includes a form of the exponential recommended in computational contexts, the scaling and squaring approach with Pade approximation (Higham, 2009), which has been implemented in ctsem accordingly.

2.2. ctsem package overview and installation

As ctsem is an R package, it requires R to be installed, available from www.r-project.org (R Core Team, 2014). The R package OpenMx (Neale et al., 2015) is required, and although it will be installed automatically via CRAN if necessary, it is recommended to download it from http://openmx.psyc.virginia.edu/, to allow use of the NPSOL optimizer. ctsem is available via CRAN, so to install and load within R simply use:

```
install.packages("ctsem")
library("ctsem")
```

For the latest development versions, http://github.com/cdriveraus/ctsem provides the Github repository, which can also be used to flag any issues noted or request support.

Estimating continuous time models via ctsem comprises four steps: First, the data must be adequately prepared (Section 2.3). Then, the continuous time model must be specified by creating a ctsem model object using the function `ctModel` (Sections 2.4 and 2.6). After specification, the model must be fit to the data using the function `ctFit`, after which `summary` and `plot` methods may be used to examine parameter estimates, standard errors, and fit statistics (Section 2.5). We will discuss these steps in the following.

2.3. Data structure

The internal functions of `ctFit` use data in a wide layout, with all data for each individual in a single row, including the time intervals between measurement occasions for this individual. Because this is the format used internally when fitting, for the sake of transparency it is also required as the input format, and is detailed below in Section 2.3.1. In some cases it may however be simpler to maintain data in a long format, and use the `ctLongToWide` and `ctIntervalise` functions we provide to convert from long format with absolute times to wide format with time intervals. This functionality is discussed in Section 2.3.2. The choice of time scale and treatment of the initial time point can influence results and will be discussed in Section 2.3.3, though first time users may find it easier to return to later.

2.3.1. Wide format

This is the data format required when fitting a model with ctsem. The example data below depicts two individuals, observed at three occasions, on three manifest variables, one time dependent predictor, and two time independent predictors. A corresponding path diagram of one possible model for this data is shown in Figure 2.1. The data are ordered into blocks as follows: Manifest process variables, time dependent predictors, time intervals, time independent predictors. Manifest variables are grouped by *measurement occasion* and ordered within this by *variable*. In the example there are three manifest variables (Y1, Y2, Y3) assessed across three measurement occasions. In this case, the first three columns of the data (Y1_T0, Y2_T0, Y3_T0) represent the three manifest variables at the first measurement occasion, time point 0, followed by the columns of the second measurement occasion and so on. Note that measurement occasions subsequent to the first may occur at different times for different individuals. Also note the naming convention, wherein the variable name is followed by an underscore and T, followed by an integer denoting the measurement occasion, beginning at T0. After the manifest variables, any time dependent predictors (there need not be any) are also grouped by *measurement occasion* and ordered within this by *variable* (changed since v2.2.0). These are named in the same way as the manifest variables, with the predictor name preceded by an underscore and T, then the measurement occasion integer beginning from 0. In the data below and the model in Figure 2.1, there is only one time dependent predictor, TD1, though more could be added. After the time dependent predictors, T-1 time intervals are specified in chronological order, with column names dT followed by the number of the measurement occasion occurring *after* the interval . That is, dT1 refers to the time interval between the first measurement occasion, T0, and the second, T1. In continuous time modelling it is imperative to know the time point at which an observation takes place. Thus, while missing values on observed scores are no problem, missing values on time intervals are not allowed. Finally, two time independent predictors (TI1, TI2 – the naming here is only with variable names) are contained in the last two columns of the data structure.

```
  Y1_T0 Y2_T0 Y3_T0 Y1_T1 Y2_T1 Y3_T1 Y1_T2 Y2_T2 Y3_T2 TD1_T0 TD1_T1 TD1_T2  dT1   dT2  TI1   TI2
1  0.44 -0.83 -0.17  1.13 -2.44  0.31    NA    NA    NA  -1.67   0.15     NA 0.65 0.001 0.06 -2.05
2    NA  6.84  9.22  8.24  9.04  7.88  6.45  8.39  7.16   0.81  -1.97   -1.6 2.56 2.260 2.22 -1.41
```

2.3.2. Conversion from long format with absolute times

Although ctsem uses the wide format as default data input, often data are stored in long format, that is, each subject has multiple rows of data, with each row reflecting a particular measurement occasion. In addition, time intervals may not be readily available at the individual level, instead the *absolute time* when a measurement took place is recorded. To convert from long format, the data must contain a subject identification column, columns for every observed variable, and a time variable. Unlike for the wide format data, at this point additional unused variables in the long structure are no problem. In the example below, three manifest variables of interest (Y1, Y2, Y3) have been observed across a number of occasions, along with one time dependent predictor (TD1) and two time independent predictors (TI1, TI2). The variable 'time' contains the time when the measurement took place (e.g., in weeks from the beginning of the study).

```
     id time    Y1    Y2    Y3   TD1   TI1   TI2
[1,]  1 0.00  5.37  6.05  7.35  2.77 -0.45 -0.23
[2,]  1   NA  5.90  3.58  7.19  1.15    NA -0.23
[3,]  1 0.89  5.92  5.05  5.09  1.55 -0.45 -0.23
[4,]  2 1.13    NA 10.77  9.57 -0.44 -0.24  1.98
[5,]  2 1.66  9.49  9.66 10.09  0.09 -0.24  1.98
```

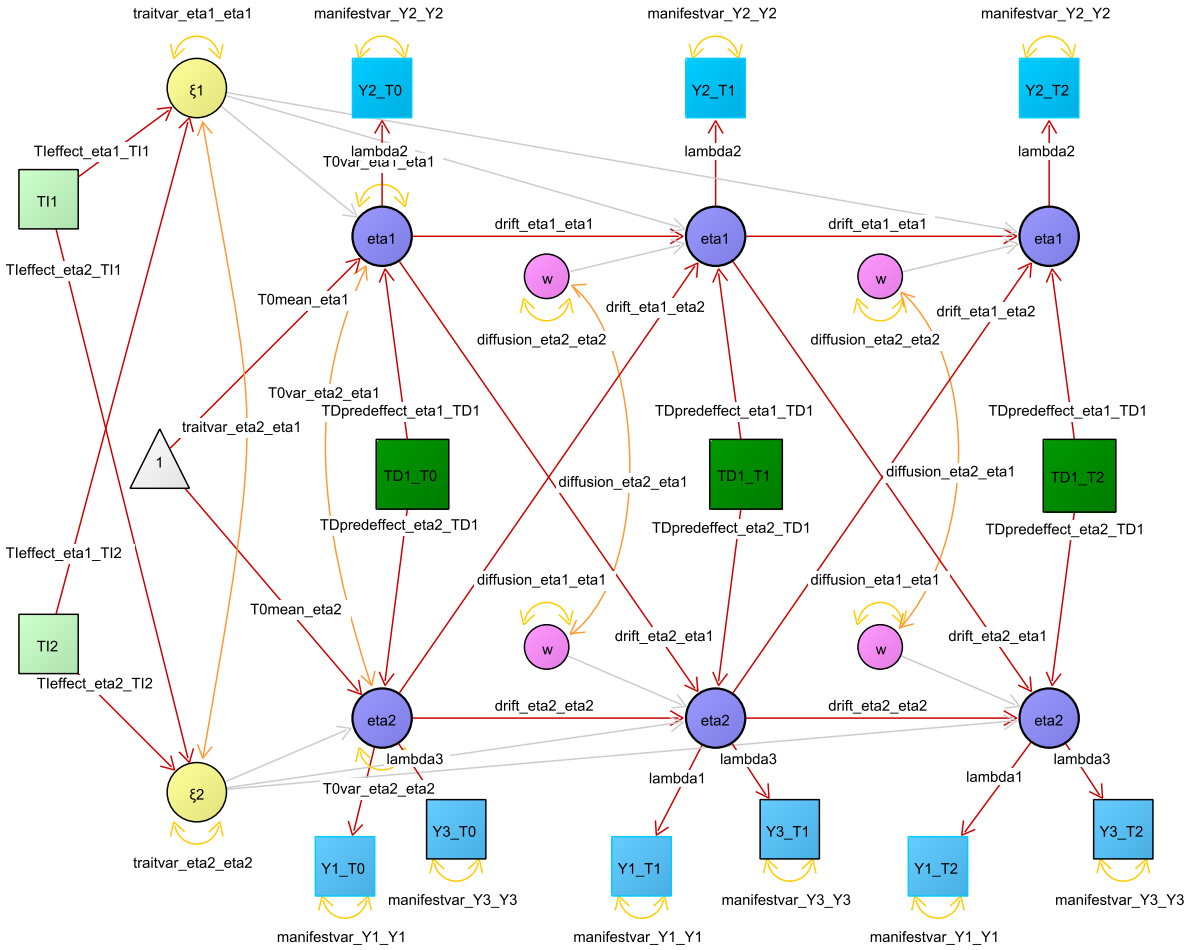


Figure 2.1.: The first three time points of a two process continuous time model, with three manifest indicators (blue) measuring 2 latent processes (purple), one time dependent predictor(dark green), and two time independent predictors (light green). Variance / covariance paths are in orange, regressions in red. Light grey paths indicate those that are constrained to a function of other parameters. Note that the value of parameters for all paths to latents at time 2 and higher do not directly represent the effect, rather, the effect depends on a function including the shown parameter and the time interval Δt. Manifest intercepts are not drawn.

```
[6,]  2 1.87  9.58  9.28  8.10  2.83 -0.24  1.98
[7,]  2 4.75 11.82  9.95  9.70 -0.72 -0.24  1.98
```

Given the specific wide structure required by ctsem, and that the time points of measurement may vary across individuals, restructuring from long to wide can be complicated, so we have included functions to manage this. First, the long format data with information on the absolute time of measurement must be converted to the wide format, using the `ctLongToWide` function (The number of Tpoints in the generated data is also messaged to the user at this point, to be used in the next step). Then, subject specific time intervals based on the absolute time information must be generated, using the function `ctIntervalise`. One should take care that the defaults used by `ctIntervalise` for structuring the data and handling missing time information are appropriate.[1]

```
data("longexample")
wideexample <- ctLongToWide(datalong = longexample, id = "id",
  time = "time", manifestNames = c("Y1", "Y2", "Y3"),
  TDpredNames = "TD1", TIpredNames = c("TI1", "TI2"))
wide <- ctIntervalise(datawide = wideexample, Tpoints = 4, n.manifest = 3,
  n.TDpred = 1, n.TIpred = 2, manifestNames = c("Y1", "Y2", "Y3"),
  TDpredNames = "TD1", TIpredNames = c("TI1", "TI2") )
```

2.3.3. Choice of initial time point and time scale

Choice of initial time point: Pre-determined or stationary?

An important aspect of continuous time modelling is the choice of how to handle the initial time point. In principle, there are two different ways to do so. One approach is to treat the first time point as *predetermined*, where no assumptions are made about the process prior to the initial time point. In this case, parameters regarding the initial latent variable (latent means and variances, and effect of predictors) are freely estimated. This is the default in ctsem, though requires some constraining if fitting a single individual.[2] When treating the first time point as predetermined, it is important to choose a meaningful starting point, as the process will gradually transition from the variances and means of the initial parameters, towards those of the parameters when the model is stationary. In principle, the initial time point does not have to reflect the first measurement occasion, and can also be set to any time prior. For example it may be of interest to set T0 to the beginning of the school year, although the first measurement was only taken two weeks after start of school. This can be specified using the `startoffset` argument to `ctIntervalise`, specifying the amount of time prior to the first observed measurement occasion. The other approach is to assume a stationary model, that is, a model where the first observations are merely random instantiations of a long term process with time-invariant mean and variance expectations. Or, put another way, we assume that sufficient time has elapsed from the unobserved, hypothetical start of the process to our first measurement occasion, such that whatever the start values were, they no longer influence the process. Strictly speaking, this requires an infinite length of time, or a process that began in a stationary state. However, in some practical cases without clear trends in the data it is possible that the improvement in estimation due to the stationarity assumption outweighs related losses (this may also be tested). To implement the stationarity assumption

[1] By default, when timing information is missing, variables measured at that time are also set to NA for the individual missing the information. Once this is done the actual time of measurement no longer influences parameter estimates or likelihood, so we can set it to an arbitrary minimum interval. By default, the `minintervalargument` to `ctIntervalise` is set to .001. This argument must be set *lower* than the minimum time interval recorded in the data, so that later observations can be adjusted without problems.

[2] Either T0VAR or T0MEANS must be fixed, see Section 2.6.3.

the means and variances of the first measurement occasion are constrained according to the model predicted means and variances across all time points. This is specified by including a character vector of the T0 matrices to constrain in the `ctFit` arguments: `stationary = c("T0VAR", "T0MEANS")` constrains both means and variances to stationarity. The `ctModel` specification of any matrices that are constrained to stationarity is ignored. Note that any between-subject variance parameters, factor loadings, manifest residuals, as well as drift and diffusion parameters, are inherently stationary (given the configuration of ctsem). More complex model specification within ctsem, or direct modification of the generated OpenMx model, is necessary for modelling time variability in the parameters.

Choice of time scale: Individual or sample relative time?

An additional consideration when treating the first time point as predetermined is necessary in cases of individually varying time intervals. Here, two alternatives need to be distinguished. The default option is to treat the observation times as *relative to the individual*, the other is to treat them as *relative to the sample*. When we treat time as relative to the individual, the first observation of every individual is set to measurement occasion T0, even though different individuals may have been recorded many years apart. However if we treat time as relative to the sample, every individual's observation times are set relative to the very first observation in the entire sample. This may result in a larger and sparser data matrix, potentially with only a single observation at the first measurement occasion. To specify sample relative time when converting from absolute time to intervals, set the argument `individualRelativeTime = FALSE` in the `ctIntervalise` function. The choice between the individual or sample relative time may influence parameter estimates when the processes are not stationary. One way of deciding between the two may be to observe whether the changes of the individuals' processes is more closely aligned with the sample relative or individual relative time. The change in processes may be more aligned with individual relative time when we expect that the activity of measurement relates to changes in the process. Consider for instance the relation between abstinence behaviour and mood among individuals attending an alcohol addiction clinic. Different individuals may come and go from the clinic over many years, but the mean level of abstinence is likely related to when each individual began attending the clinic and being measured – not the specific date the observation took place. In contrast, sample relative time could be more appropriate for a study of linguistic abilities in a cohort of schoolchildren over the years, with some individuals observed early and some only observed later, once they are older and more developed. In this case, we may expect changes in the average linguistic ability related to sample time. Another example that becomes conveniently available with continuous time models and these functions is to arrange the data in individual relative fashion but using age as the timing variable. In this case, age-related developmental trajectories may be studied. When considering these options one should be aware that consistent up or down trends over time may confound dynamic parameter estimates, if the innovation (latent residual) at t is correlated with the process at $t-1$. Pre-processing approaches that remove trend components, such as controlling for age or year, removing a linear trend, or differencing scores, *may* provide some check on model estimates, but the ramifications of these should be carefully considered. Alternatively one may wish to explicitly model the diffusion process, discussed in Section 2.6.5.

2.4. Model specification

Continuous time models are specified via the `ctModel` function. This function takes as input a series of arguments and parameter matrices, and outputs a list object containing matrices to be later evaluated by the ctFit function. The `ctModel` function contains many defaults that should be generally applicable and safe, in that most parameters are specified to be freely

estimated, with a few exceptions.[3] However, as with all default settings, they should be checked as they may not be applicable. The arguments to the `ctModel` function and the relation to equations in Section 2.1 are shown in Table 2.1 (required specification) and Table 2.2 (optional specification). The matrices can be specified with either character labels, to indicate free parameter names, or numeric values, which indicate fixed values. A mixture of both in one matrix is fine. These generally need to be set when constraining parameters to equality (same character label), when fixing certain parameters to specific values (for instance, when you do not wish to have a certain parameter in the model, or when testing if an effect is different from 0), or when assigning non-standard names to output parameters.

An example model specification relying heavily on the defaults is:

```
examplemodel <- ctModel(n.latent = 2, n.manifest = 2, Tpoints = 3, LAMBDA = diag(2))
```

A visual representation of this model is shown in Figure 2.2. With `n.latent = 2`, we have specified a model with 2 latent processes, shown in purple. Each of these is measured by a single manifest indicator (in blue), for a total of 2 manifest variables, specified with `n.manifest = 2`. Loadings between latents and manifests are fixed to 1.00 (indicated by the 2×2 diagonal LAMBDA matrix) at 3 measurement occasions, specified by `Tpoints = 3`. Because no other parameters are specified, the model defaults are used, resulting in a bivariate latent process model where each manifest variable has a measurement error variance (manifestvar_Y1_Y1, manifestvar_Y2_Y2), and an estimated intercept (not shown). The initial latent variables of each process have freely estimated means (T0mean_eta1, T0mean_eta2), variances (T0var_eta1_eta1, T0var_eta2_eta2), and covariance (T0var_eta2_eta1). Subsequent latent variables of each process all have an innovation term, with the variance dependent on a function of the diffusion matrix (variances diffusion_eta1_eta1, diffusion_eta2_eta2, covariance diffusion_eta2_eta1), drift matrix, and time interval Δt (Note that although we speak here of variance and covariance parameters for the sake of intuitive understanding, ctsem works with Cholesky decomposed covariance matrices, discussed in Section 2.4). Each latent variable in our two processes has continuous auto effects on itself according to the drift_eta1_eta1 and drift_eta2_eta2 parameters (the diagonals of the drift matrix), and cross effects to the other process according to the drift_eta1_eta2 and drift_eta2_eta1 parameters (the off diagonals). This drift matrix combines with time interval Δt to generate the auto and cross regressions shown in the diagram. As usual, the first process listed in the parameter name represents the row of the drift matrix, and the second the column, with the direction of effects flowing from column to row – so the parameter drift_eta1_eta2 represents the effect of a change in process 2 on later values of process 1. Each process also has a continuous intercept (cint_eta1, cint_eta2), which, in combination with the drift matrix, sets the level to which each process asymptotes. By default these are fixed to zero (and hence not shown, as the manifest means and continuous intercept parameters cannot all be free, or the model is not identified). To develop an understanding of the parameter matrices or simply view a model, printing the model object (e.g. `print(examplemodel)`) is recommended. To track how these matrices are used within the complete SEM specification, one must first estimate the model (discussed in Section 2.5), and may then view the A, S, F or M matrices typical to a RAM specification J. Jack McArdle and McDonald (1984) via `example1fit$mxobj$A` (for the A matrix).

[3] `ctModel` defaults that *may* not be considered safe, as they are not freely estimated by default, are the TRAITVAR and MANIFESTTRAITVAR matrices. While it is very likely that with multiple subjects one or the other matrix will need to be freed, only one of the two trait matrices can be set at once. See Section 2.6.1 regarding the trait matrices.

Argument	Sign	Meaning
n.manifest	c	Number of manifest indicators per individual at each measurement occasion.
n.latent	v	Number of latent processes.
Tpoints		Number of time points, or measurement occasions, in the data.
LAMBDA	Λ	n.manifest × n.latent loading matrix relating latent to manifest variables.

Table 2.1.: Required arguments for ctModel.

Argument	Sign	Default	Meaning.
manifestNames		Y1, Y2, etc	n.manifest length character vector of manifest names.
latentNames		eta1, eta2, etc	n.latent length character vector of latent names.
T0VAR		free	lower triangular n.latent × n.latent Cholesky matrix of latent process initial variance / covariance.
T0MEANS		free	n.latent × 1 matrix of latent process means at first time point, T0.
MANIFESTMEANS	Γ	0	n.manifest × 1 matrix of manifest means.
MANIFESTVAR	Θ	free diag	lower triangular n.manifest × n.manifest Cholesky matrix of variance / covariance between manifests (i.e., measurement error).
DRIFT	$\mathbf{A}$	free	n.latent × n.latent matrix of continuous auto and cross effects.
CINT	$\boldsymbol{\kappa}$	free	n.latent × 1 matrix of continuous intercepts.
DIFFUSION	$\mathbf{Q}$	free	lower triangular n.latent × n.latent Cholesky matrix of diffusion variance / covariance.
TRAITVAR	Υ	NULL	NULL if no trait variance, or lower triangular n.latent × n.latent Cholesky matrix of trait variance / covariance.
MANIFESTTRAITVAR	Ψ	NULL	NULL if no trait variance on manifest indicators, or lower triangular n.manifest × n.manifest Cholesky matrix.
n.TDpred	l	0	Number of time dependent predictors in the dataset.
TDpredNames		TD1, TD2, etc	n.TDpred length character vector of time dependent predictor names.
TDPREDMEANS		free	n.TDpred × Tpoints rows × 1 column matrix of time dependent predictor means.
TDPREDEFFECT	$\mathbf{M}$	free	n.latent × n.TDpred matrix of effects from time dependent predictors to latent processes.
T0TDPREDCOV		0	n.latent × (Tpoints × n.TDpred) covariance matrix between latents at T0 and time dependent predictors.
TDPREDVAR		free	lower triangular (n.TDpred × Tpoints) × (n.TDpred × Tpoints) Cholesky matrix for time dependent predictors variance / covariance.
TRAITTDPREDCOV		0	n.latent rows × (n.TDpred × Tpoints) columns covariance matrix for latent traits and time dependent predictors.
TDTIPREDCOV		0	(n.TDpred × Tpoints) rows × n.TIpred columns covariance matrix between time dependent and independent predictors.
n.TIpred	p	0	Number of time independent predictors.
TIpredNames		TI1, TI2, etc	n.TIpred length character vector of time independent predictor names.
TIPREDMEANS		free	n.TIpred × 1 matrix of time independent predictor means.
TIPREDEFFECT	$\mathbf{B}$	free	n.latent × n.TIpred effect matrix of time independent predictors on latent processes.
T0TIPREDEFFECT		free	n.latent × n.TIpred effect matrix of time independent predictors on latents at T0.
TIPREDVAR		free	lower triangular n.TIpred × n.TIpred Cholesky matrix of time independent predictors variance / covariance.
startValues		NULL	a named vector, where the names of each value must match a parameter in the specified model, and the value sets the starting value for that parameter during optimization.

Table 2.2.: Optional arguments for ctModel.

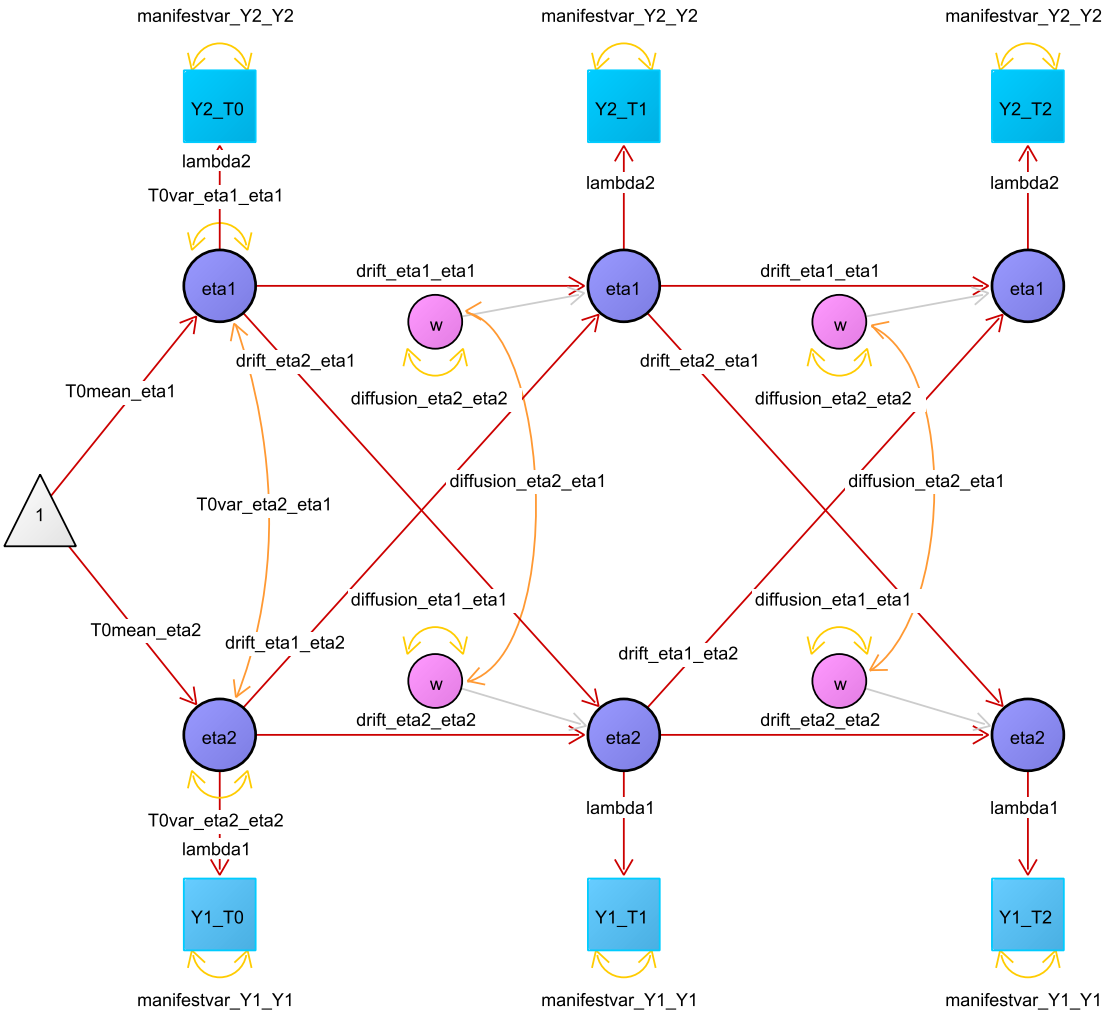


Figure 2.2.: A two process continuous time model with manifest indicators (blue) measuring latent processes (purple). Variance / covariance paths are in orange, regressions in red. Light grey paths indicate those that are either fixed to certain values or constrained to other parameters. Note that the value of the parameters for all paths to latents at time 2 and higher do not directly represent the effect, rather, the effect depends on a function of the shown parameter and the time interval Δt. This model includes neither observed or unobserved between person variance, nor any time dependent predictors.

Cholesky decomposed variance / covariance input matrices

To ensure reliable estimation, some parameter transformations have been implemented in ctsem for the estimation of covariance matrices. Rather than directly operate on covariance matrices, ctsem takes as input Cholesky decomposed covariance matrices, as these allow for unbounded estimation. The Cholesky decomposition is such that variance / covariance matrix $\Sigma = LL^\top$, where L is lower-triangular. This means that input variance / covariance matrices for ctsem must be lower triangular. The meaning of a 0 in the matrix is the same for both covariance and Cholesky decomposition approaches. An important point to be aware of is that while Cholesky matrices are required as input, for convenience, the matrices reported in the `summary` function are full variance / covariance matrices. These can be converted to the Cholesky decomposed form using code in the form `t(chol(summary(ctfitobj)$varcovmatrix))`.

While not affecting interpretations of the matrix input or output, internally, by default ctsem also optimizes over the natural logarithm of the diagonal of the Cholesky factor covariance matrices. These transformations are reflected in the raw OpenMx parameter output section of the output summary (when `verbose=TRUE`), but otherwise require no specific knowledge or action – the logarithmic transformations take place internally, and the regular variance / covariance matrices are displayed in the summary matrices.

2.5. Model estimation

The `ctFit` function estimates the specified model, calling the data in wide format along with the ctsem model object. For an example, we can fit a similar model to that defined in Section 2.4. We first load an example dataset contained in the ctsem package, then use the `ctFit` function for parameter estimation. Output information can be obtained via the `summary` function. The dataset used in this example, is a simulation of the relation between leisure time and happiness for 100 individuals across 6 measurement occasions. Because our data here does not use the default manifest variable names of `Y1` and `Y2`, but rather `LeisureTime` and `Happiness`, we must include a `manifestNames` character vector in our model specification. Because each manifest directly measures a latent process, we can use the same character vector for the `latentNames` argument, though one could specify any character vector of length 2 here, or rely on the defaults of `eta1` and `eta2`.

```
data("ctExample1")
example1model <- ctModel(n.latent = 2, n.manifest = 2, Tpoints = 6,
  manifestNames = c("LeisureTime", "Happiness"),
  latentNames = c("LeisureTime", "Happiness"), LAMBDA = diag(2))
example1fit <- ctFit(datawide = ctExample1, ctmodelobj = example1model)
```

The output of `summary` after fitting such a model includes the matrices representing the continuous time parameters (e.g., DRIFT), a list of estimates of only the free parameters, and fit information from the `OpenMx` summary function. Further information can be obtained using the argument `verbose=TRUE`, which will return the raw OpenMx parameter values and standard errors, as well as additional summary matrices of discrete time transformations for the time interval $\Delta t = 1$ (e.g., discreteDRIFT), and when appropriate, asymptotic values for the parameters as the time interval Δt approaches ∞ (e.g., asymDIFFUSION may be taken to represent the total within subject variance of a process). When appropriate, standardised matrices are also output with the suffix 'std'.[4]

[4]Standardisations are based on only the relevant variance, not the total. For instance, DRIFT parameters are standardised using only the within-subject variance, asymDIFFUSION, because DRIFT parameters are typically intended to represent individual, or average individual, temporal dynamics.

```
summary(example1fit, verbose = TRUE)["discreteDRIFTstd"]
$discreteDRIFTstd
            LeisureTime   Happiness
LeisureTime  0.97275721 -0.04989875
Happiness   -0.01382697  0.91462401
```

The output above shows the standardised discrete time equivalent of the DRIFT matrix for time interval $\Delta t = 1$. This is provided for convenience, but one should note that it only represents the temporal effects given the specific interval of 1 unit of time (The specific interval shown for the discrete summary matrices may be modified with the argument `timeInterval`). The unstandardised discreteDRIFT matrix may be calculated from the continuous drift matrix for any desired interval. The following code shows this calculation for a time interval of 2.5:

```
expm(summary(example1fit)$DRIFT * 2.5)
```

From the diagonals of the discreteDRIFTstd matrix we see that changes in the amount of leisure time one has tend to persist longer (indicated by a higher autoregression) than happiness. The cross-regression in row 2 column 1 suggests that as leisure time increases, this tends to be followed by *decreases* in happiness. Similarly, the cross-regression in row 1 column 2 suggests that as happiness increases, this tends to be followed by *decreases* in leisure time. While these results are accurate for the specified model, the specified model is likely inappropriate for this data, which we explain more of in Section 2.6.1 on unobserved heterogeneity.

2.5.1. Comparing different models

Suppose we wanted to test the model we fit above against a model where the effect of happiness on later leisure time (parameter drift_LeisureTime_Happiness) was constrained to 0. First we specify and fit the model under the null hypothesis by taking our previous model and fixing the desired parameter to 0:

```
testmodel <- example1model
testmodel$DRIFT[1, 2] <- 0
testfit <- ctFit(datawide = ctExample1, ctmodelobj = testmodel)
```

The result may then be compared to the original model with a likelihood ratio test, using the OpenMx function `mxCompare`. To use this function a base model fit object and a comparison model fit object must be specified, with the latter being a constrained version of the former. Note that ctsem stores the original OpenMx fit object under a `$mxobj` sub-object, which must be referenced when using OpenMx functions directly.

```
mxCompare(example1fit$mxobj, testfit$mxobj)
   base comparison ep minus2LL   df      AIC   diffLL diffdf           p
1 ctsem       <NA> 16 4176.875 1184 1808.875       NA     NA          NA
2 ctsem      ctsem 15 4196.735 1185 1826.735 19.86085      1 8.32885e-06
```

According to the conventional p ¡ .05 criterion, results show that the more constrained model fits the data significantly worse, that is, happiness has a significant effect on later leisure time for this model and data. An alternative to this approach is to estimate 95% profile-likelihood confidence intervals for our parameters of interest, from our already fit model:

```
example1cifit <- ctCI(example1fit, confidenceintervals = "DRIFT")
                                  lbound    estimate       ubound note
drift_LeisureTime_LeisureTime -0.04680600 -0.02800068 -0.012462498
drift_LeisureTime_Happiness   -0.10828585 -0.06974154 -0.037744354
drift_Happiness_LeisureTime   -0.03116298 -0.01112155  0.008695662
drift_Happiness_Happiness     -0.14855570 -0.08963807 -0.045938702
```

Now the `summary` function reports 95% confidence bounds for the continuous drift parameters, which in case of drift_LeisureTime_Happiness (DRIFT[1,2]) does not include 0. For complicated models, the estimation of confidence intervals may increase computation time considerably. One could also compute a confidence interval by multiplying the standard error of the estimate (returned in the summary) by 1.96, however profile-likelihood confidence intervals are in general recommended as they do not assume symmetric intervals. We have observed however that optimization difficulties can sometimes result in inacccurate (extremely close to the point estimate, accuracy can be checked in the lower and upper delta returned by `example1cifit$mxobj$intervals` subobject) or missing profile-likelihood confidence intervals, in particular when numbers of subjects or time points is limited, and in general standard error based intervals appear to perform quite reasonably.

2.5.2. Plots

A visual depiction of the relationships between the processes over time is given by the `plot.ctsemFit` (which can be called simply by `plot`) function for any fit object created by `ctFit`. Depending on arguments, this function can show the processes' mean trajectories, within-subject variance, autoregression, and cross regression plots, as well as plots showing expected changes in each process given either an *observed* change of 1.00, or an *exogenous input* of 1.00 (The former is a mixture of the DIFFUSION and DRIFT matrices, while the latter is just an alternative representation of the auto and cross regression plots). Autoregression plots show the impact of a 1 unit change in a process on later values of that process, while cross regression plots show the impact of a 1 unit change in one process on later values of other processes. Some examples can be seen in Figure 2.3.

2.6. Continuous time models: extensions

2.6.1. Unobserved heterogeneity

Traits at the latent level

When modelling panel data, the continuous intercept parameter $\boldsymbol{\kappa}$ reflects the expected value for continuous time intercept b, which determines the average level of a process ($\boldsymbol{\kappa}$ is fixed to 0 in ctsem by default, as free manifest means account for non zero equilibrium levels in the data). In panel data, however, it is common that *individuals* exhibit stable differences in the level. Within ctsem we call such stable differences *traits*, but they may also be thought of more abstractly as *unit level* or *between subject* differences, or *unobserved heterogeneity*. Fitting a model that fails to account for it will result in parameter estimates that will not reflect the processes of individual subjects, but will mix between and within-person information (Balestra & Nerlove, 1966; Oud & Jansen, 2000; Halaby, 2004). To avoid this bias, individual differences can be incorporated in two different ways. One way is to control for observed covariates as will be discussed in Section 2.6.2. However as covariates are likely to be insufficient, one may also estimate the latent trait variance by estimating the variance and covariance Υ of the continuous intercept parameters b across individuals.[5] In ctsem, freely estimated latent trait variances and covariances may be added with the argument `TRAITVAR = "auto"` to the `ctModel` command. If the user is interested in a specific variance-covariance structure, it is of course also possible to specify the n.latent × n.latent lower-triangular matrix of free or fixed parameters by hand.

[5] Note that this is a substantially different approach to achieve unbiased effect estimates than the *fixed effects* approach (see for example Mundlak, 1978), as our SEM specification, while a *random effects* model which have at times been associated with bias for within effects, allows unbiased estimation of *within* and *between* effects at the same time. For further details on the estimation of unobserved heterogeneity in an SEM context, see Bollen and Brand (2010).

To illustrate the inclusion of trait variance, we fit the same model on simulated leisure time and happiness introduced above, but also model the traits.

```
data("ctExample1")
traitmodel <- ctModel(n.manifest = 2, n.latent = 2, Tpoints = 6,
  LAMBDA = diag(2), manifestNames = c("LeisureTime", "Happiness"),
  latentNames = c("LeisureTime", "Happiness"), TRAITVAR = "auto")
traitfit <- ctFit(datawide = ctExample1, ctmodelobj = traitmodel)
```

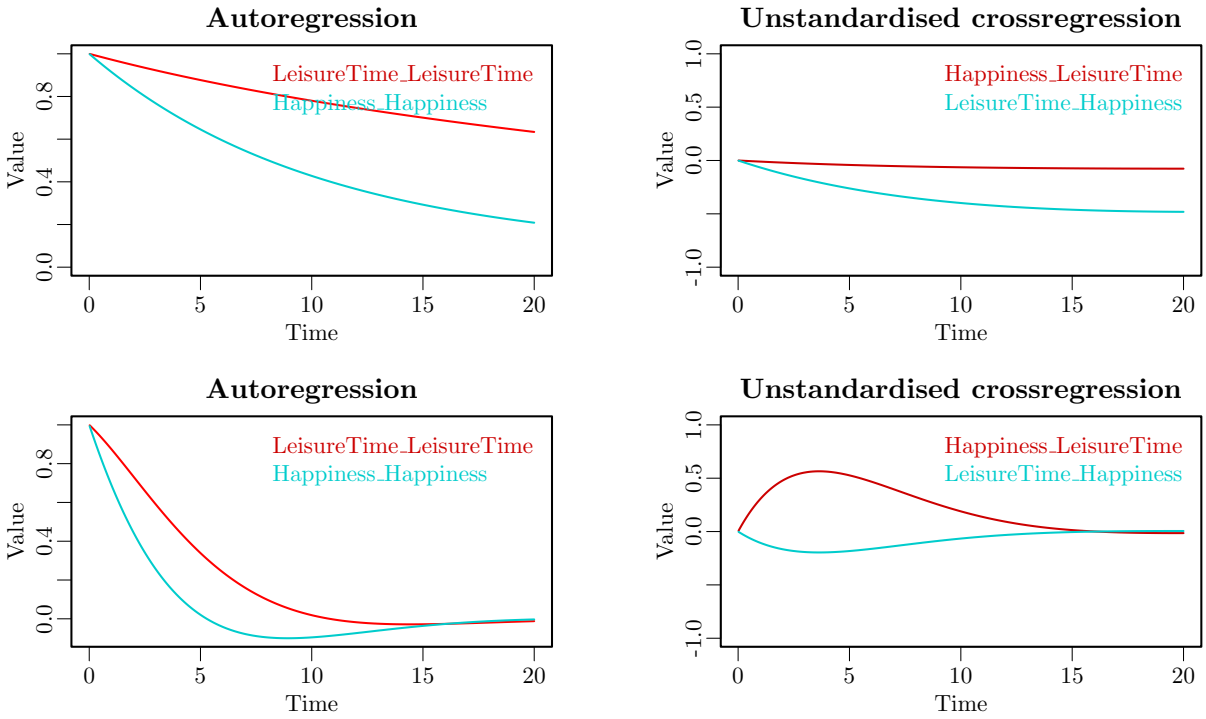


Figure 2.3.: Top row shows parameter plots without accounting for trait variance, bottom row with trait variance accounted for. Plots show how auto and cross effects change depending on the length of time between observations.

From Figure 2.3, we can see that after accounting for differences in the trait levels of leisure time and happiness, the estimated auto and cross regression effects between latent processes are very different. Auto effects (persistence) have reduced, and the magnitude and sign of the cross effects have switched. Now, rather than a *decrease in leisure time* predicting an *increase in happiness*, after controlling for unobserved heterogeneity we see instead that *increases in leisure time* predict later *increases in happiness*.

Traits at the indicator level

Beyond differences in the level of the latent process, it is also possible that stable individual differences in the level of some or all indicators of a process may exist, and as such may be better accounted for at the measurement level. Take for instance a latent process, happiness, estimated using three survey questions at 10 time points for multiple individuals. According to the models we have described so far, the estimated manifest means apply equally to all individuals. However, consider that question three queries happiness with work, which may for some people be consistently high, independent of their actual latent happiness, and for some may be consistently low. Calculating the latent process using the same mean for happiness with work again confounds between and within person information, but we can account for

this by using what we will refer to as *manifest traits* – an additional, time invariant variance-covariance structure on the measurement level. These are specified by including the `MANIFESTTRAITVAR` matrix in the `ctModel` specification, either as `MANIFESTTRAITVAR = "auto"` wherein time invariant variance and covariance for all indicators is freely estimated, or the n.manifest × n.manifest lower-triangular matrix can be specified explicitly as usual. Such a specification may allow for improved fit of factor models, more realistic estimates of the dynamics of individual processes, and the testing of measurement related hypotheses. Note however that identifying restrictions will be necessary for any model that contains both manifest and process level traits – one possible form for this may be a free process level TRAITVAR matrix and a MANIFESTTRAITVAR matrix that is fixed to 0 across factors, but free within any factors that are measured by more than one indicator.

2.6.2. Predictors

ctsem allows the inclusion of *time independent* as well as time *time dependent* exogenous predictors. Time independent predictors could be variables such as gender, personality or socio-demographic background variables that remain constant over time. An example of a time dependent predictor could be a financial crisis, which all individuals in the sample experience at the same time, or the death of a loved one, which only some individuals may experience and for whom the time point of the event may differ. Both events may be thought of as adding some relatively distinct and sudden change to an individual's life, which influences the processes of interest. Time dependent predictors are distinguished from the endogenous latent processes in that they are assumed to be independent of fluctuations in the processes – changes in the latent processes do not lead to changes in the predictor. Furthermore, no temporal structure between different time points is modeled. Because of these two assumptions, in any case where the time dependent predictor depends on earlier values of either itself or the latent process, it may be better to model it as an additional latent process.

Time independent predictors

Time independent predictors are added by including the data as per the structures shown in Section 2.3, and specifying the number of time independent predictors, `n.TIpred`, in the `ctModel` arguments. If not using the default variable naming, a `TIpredNames` character vector should also be specified. For an example, we add the 'number of close friends' as a time independent predictor to the earlier leisure time and happiness model. Note that, just like in any conventional regression analysis, if time independent predictors are not centered around 0, the estimate of continuous intercept parameters depends on the mean of the predictor.

```
data("ctExample1TIpred")
tipredmodel <- ctModel(n.manifest = 2, n.latent = 2, n.TIpred = 1,
  manifestNames = c("LeisureTime", "Happiness"),
  latentNames = c("LeisureTime", "Happiness"),
  TIpredNames = "NumFriends",
 Tpoints = 6, LAMBDA = diag(2), TRAITVAR = "auto")
tipredfit <- ctFit(datawide = ctExample1TIpred, ctmodelobj = tipredmodel)

summary(tipredfit, verbose = TRUE)["TIPREDEFFECT"]
summary(tipredfit, verbose = TRUE)["discreteTIPREDEFFECT"]
summary(tipredfit, verbose = TRUE)["asymTIPREDEFFECT"]
summary(tipredfit, verbose = TRUE)["addedTIPREDVAR"]
```

```
$TIPREDEFFECT
             NumFriends
LeisureTime -0.2254703
Happiness    0.5488260

$asymTIPREDEFFECT
             NumFriends
LeisureTime -1.6726366
Happiness    0.2192087

$discreteTIPREDEFFECT
             NumFriends
LeisureTime -0.2386960
Happiness    0.4416718

$addedTIPREDVAR
             LeisureTime   Happiness
LeisureTime    2.8375087 -0.37187192
Happiness     -0.3718719  0.04873596
```

The matrices output from `summary, verbose = TRUE` will now include matrices related to time independent predictors, while the estimated parameters now also includes a range of variance, covariance, and effect parameters for time independent predictors. Matrix TIPREDEFFECT displays the continuous time effect parameters, however discreteTIPREDEFFECT shows the effect added to the processes for each unit of time, which may provide a useful comparison with discrete time models. asymTIPREDEFFECT (Asymptotic time independent predictor effect) shows the expected total change in process means given an increase of 1 on the time independent predictor. From these matrices we see that the number of close friends has a negative relationship to leisure time, but a positive relationship to happiness. The final matrix, addedTIPREDVAR, displays the stable between-subject variance and covariance in the processes accounted for by the time independent predictors.

Time dependent predictors

ctsem allows the specification of time dependent predictors: The fundamental form of such a predictor is that of a sudden impulse to the system which then dissipates back to the process mean, however with some thought it is possible to specify a wide range of effect shapes. Figure 2.4 provides an example of two different extremes, the basic impulse form and a permanent level change form.

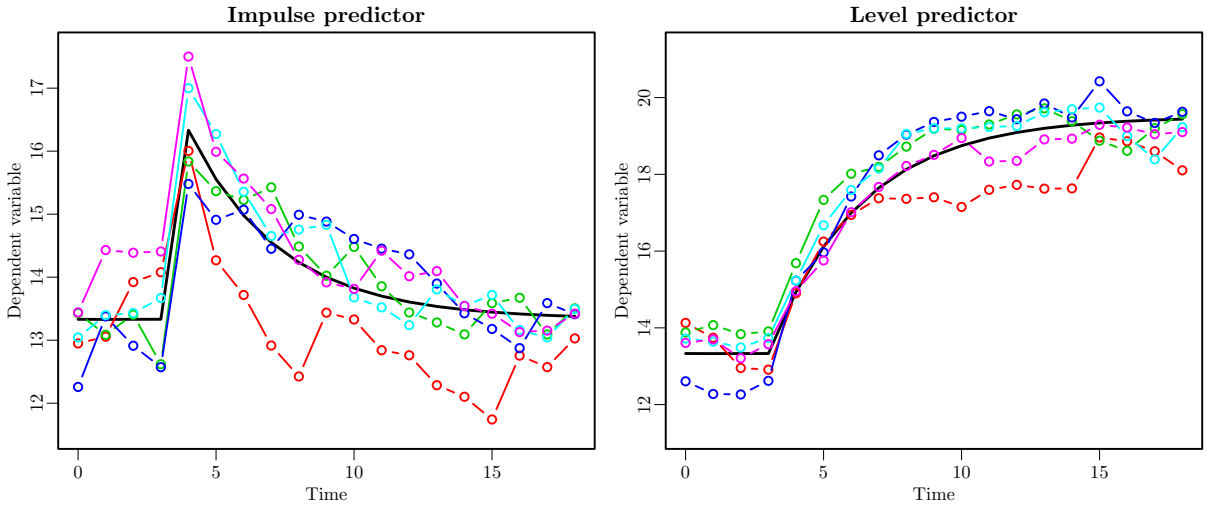


Figure 2.4.: Two shapes of time dependent predictors: both plots show 5 selected individuals data, all experiencing a time dependent predictor at time point 5. The model-based expected trajectory of the predictor effect (including autoregression) is also shown as a solid black line. On the left, the processes spike up and then dissipate, reflecting a transient change, or *impulse*. On the right, the processes trend upwards towards a new equilibrium, reflecting a stable change in the *level*.

A single time dependent predictor can be incorporated in a ctsem model by adding the argument `n.TDpred = 1` to the `ctModel` function, as well as a `TDpredNames` vector if not using the default variable naming in your data, then fitting as usual. In the following example, we use the same two simulated processes as above and include an intervention that all individuals experience at time 5. For example, let us assume everyone receives a large amount of money and we are interested in the impact of this monetary gift on leisure time and happiness. We expect that some short term increase in both leisure time and happiness may occur, as people may take holidays or enjoy the unexpected boon otherwise, but we also want to check whether the gift we provide may also cause a longer term adjustment in leisure time or happiness. To this end we first fit a model with the basic impulse effect, coded in the data as a 1 when the intervention occurs and a 0 otherwise.[6]

```
data("ctExample2")
tdpredmodel <- ctModel(n.manifest = 2, n.latent = 2, n.TDpred = 1,
  Tpoints = 8, manifestNames = c("LeisureTime", "Happiness"),
  TDpredNames = "MoneyInt", latentNames = c("LeisureTime", "Happiness"),
  LAMBDA = diag(2), TRAITVAR = "auto")
tdpredfit <- ctFit(datawide = ctExample2, ctmodelobj = tdpredmodel)

summary(tdpredfit, verbose = TRUE)["TDPREDEFFECT"]
$TDPREDEFFECT
             MoneyInt
LeisureTime 0.4119485
Happiness   0.6963775
```

The matrices reported from `summary(tdpredfit, verbose = TRUE)` will now include those related to the time dependent predictor, and the parameters section will include all the additional free parameters estimated, including many variance and covariance related parameters, and the effect parameters TDpred_LeisureTime_MoneyInt and TDpred_Happiness_MoneyInt. Looking at the summary matrices, TDPREDEFFECT shows us the initial impact of the predictor on the processes. From the matrices, we can see that the monetary intervention relates directly to subsequent increases in both leisure time and happiness. Standardised estimates are not provided because we assume no model for the variance of time dependent predictors.

Adding a level change predictor

To test the longer term changes introduced via the monetary intervention, we must model the impact of the predictor via an intermediate latent process: We fix the intercepts (T0MEANS and CINT) and random variance (T0VAR, DIFFUSION, and TRAITVAR) of this additional process to 0; set changes to persist indefinitely via a diagonal DRIFT value very close to 0 (precisely 0 causes computational problems); fix the impact of the predictor on the new process to 1 (to identify the effect); fix the impact of the two original latent processes on the new process to 0 (via the off-diagonals in the third row of DRIFT); and estimate the impact of the additional process on our original two processes of interest (via the off-diagonals in the third column of DRIFT). Alternatively, one could also estimate the time course of predictor effects by freeing the DRIFT diagonal of the additional process.

```
data("ctExample2")
tdpredlevelmodel <- ctModel(n.manifest = 2, n.latent = 3,
  n.TDpred = 1,
  Tpoints = 8, manifestNames = c("LeisureTime", "Happiness"),
  TDpredNames = "MoneyInt",
```

[6] While this form of dummy coding works well, if there are predictors with no variance and the TDPREDVAR matrix is not specified, ctsem warns the user and fixes TDPREDVAR to a diagonal matrix with small variance.

```
  latentNames = c("LeisureTime", "Happiness", "MoneyIntLatent"),
  LAMBDA = matrix(c(1,0, 0,1, 0,0), ncol = 3), TRAITVAR = "auto")

tdpredlevelmodel$TRAITVAR[3, ] <- 0
tdpredlevelmodel$TRAITVAR[, 3] <- 0
tdpredlevelmodel$DIFFUSION[, 3] <- 0
tdpredlevelmodel$DIFFUSION[3, ] <- 0
tdpredlevelmodel$T0VAR[3, ] <- 0
tdpredlevelmodel$T0VAR[, 3] <- 0
tdpredlevelmodel$CINT[3] <- 0
tdpredlevelmodel$T0MEANS[3] <- 0
tdpredlevelmodel$TDPREDEFFECT[1:3, ] <- c(0,0,1)
tdpredlevelmodel$DRIFT[3, ] <- c(0,0,-.000001)

tdpredlevelfit <- ctFit(datawide = ctExample2,
  ctmodelobj = tdpredlevelmodel)

summary(tdpredlevelfit, verbose = TRUE)[c("DRIFT","TDPREDEFFECT")]
$DRIFT
               LeisureTime   Happiness MoneyIntLatent
LeisureTime    -0.13934880 -0.03936504      0.5699073
Happiness       0.07981293 -0.10377863     -0.3576735
MoneyIntLatent  0.00000000  0.00000000     -0.0000010

$TDPREDEFFECT
               MoneyInt
LeisureTime           0
Happiness             0
MoneyIntLatent        1
```

Now, if we look at column 3 of the DRIFT matrix, we see that the monetary intervention process appears to cause long term increases in leisure time, but potentially reductions in happiness.

2.6.3. $N = 1$ time series with multiple indicators

In the examples so far, we have dealt with multiple individuals with relatively few measurement occasions, and latent processes have been estimated by a single indicator. However, ctsem may also be used for the analysis of time series data for single subjects observed at many measurement occasions, as well as the estimation of latent factors estimated from multiple indicators. With single-subject data, a Kalman filter implementation is typically far quicker than the matrix arrangement we use for multiple subjects, however ctsem allows either to be used. To illustrate these features, we perform a dynamic factor analysis on a single individual, with three manifest indicators measured at 50 occasions. Because the model is fitted to a single individual, we cannot freely estimate both the latent variance and mean at the first measurement occasion, but we must fix the 1×1 `T0VAR` matrix to a reasonable value, or implement stationarity constraints as discussed in Section 2.3.3. The precise fixed value becomes unimportant as the time series length increases (Durbin & Koopman, 2012). Note that in this example the LAMBDA matrix specifies a loading of 1.00 for manifest Y1 (for identification), while loadings for Y2 and Y3 are freely estimated. Note also that although ctsem uses the Kalman filter by default when a single subject is specified, this can be overridden by specifying the `objective = "mxRAM"` argument to `ctFit`, if one wishes to use the slower RAM implementation. The Kalman filter may also be specified for multiple subjects. In this case, between subject trait or time independent predictor matrices are ignored – one may need to account for consistent differences between subjects through pre-processing or thoughtful expansion of the state matrices.

```
data("ctExample3")
model <- ctModel(n.latent = 1, n.manifest = 3, Tpoints = 100,
  LAMBDA = matrix(c(1, "lambda2", "lambda3"), nrow = 3, ncol = 1),
  MANIFESTMEANS = matrix(c(0, "manifestmean2", "manifestmean3"), nrow = 3,
    ncol = 1))
fit <- ctFit(data = ctExample3, ctmodelobj = model, objective = "Kalman",
  stationary = c("T0VAR"))
```

2.6.4. Multiple group continuous time models

In some cases, certain groups or individuals may exhibit different model parameters. We can investigate group or individual level differences by specifying a multiple group model using the `ctMultigroupFit` function. For this example, we will use the same model structure as in the single subject example from Section 2.6.3, but apply it to two groups of 10 individuals, whom we expect to exhibit differences in the loading of the third manifest variable. When using `ctMultigroupFit`, all parameters are free across groups by default. However, in addition to the standard model specification you may also specify either a *fixed model*, or a *free model*. A fixed model should be of the same structure as the base model, with any parameters you wish to constrain across groups set to the character string 'groupfixed'. The value for any other parameters is not important. Alternatively, one may specify a free model, where any parameters to freely estimate for each group are given the label 'groupfree', and all others will be constrained across groups. In this example, because we only want to examine group differences on one parameter, we specify a free model in which the loading parameter between manifest3 and our latent process eta1 is labelled 'groupfree' – this estimates distinct lambda3 parameters for each group, and constrains all other parameters across the two groups to equality. The group specific parameter estimates will appear in the resulting summary prefixed by the specified *grouping vector*. This is the final requirement for `ctMultigroupFit` and is simply a vector specifying a group label for each row of our data. In this case we have groups one and two, containing the first and the last 10 rows of data respectively, prefixed by the letter 'g' to denote group.

```
data("ctExample4")

basemodel <- ctModel(n.latent = 1, n.manifest = 3, Tpoints = 20,
  LAMBDA = matrix(c(1, "lambda2", "lambda3"), nrow = 3, ncol = 1),
  MANIFESTMEANS = matrix(c(0, "manifestmean2",
    "manifestmean3"), nrow = 3, ncol = 1))

freemodel <- basemodel
freemodel$LAMBDA[3, 1] <- "groupfree"
groups <- paste0("g", rep(1:2, each = 10))

multif <- ctMultigroupFit(datawide = ctExample4, groupings = groups,
  objective="Kalman", ctmodelobj = basemodel, freemodel = freemodel)



g1_lambda3 g2_lambda3
 1.4168917  0.2077628
```

Looking at the estimated parameters from the `$omxsummary` (OpenMx) portion of `summary , verbose = TRUE`, we indeed see a difference between parameters g1_lambda3 (group 1) and g2_lambda3 (group 2), and could test this with the usual approaches discussed in Section 2.5.1. A point to note is that the multiple group and Kalman filter implementations can be easily combined by specifying a distinct group for each row of data. This can allow for a mixture of individual and group level parameters.

2.6.5. Higher order models and simulating data

In the models discussed so far, the individual processes were only conceived of as first order processes, always tending to revert to baseline when away from it. However, what about a situation where we have variables which show very slow patterns of change, upwards or downwards trajectories that are maintained over many observations? This can provide for oscillations and slower patterns of change, as for example with damped linear oscillators, or moving average like effects as from the ARMA modelling framework.

Continuous time models of this variety are theoretically plausible, as changes to the level of a process are not necessarily always random in direction with a tendency to baseline, but may depend on contextual circumstances that have some persistence. Consider an individual's overall health over the course of 20 years, sampled every few months. If the individual changes exercise or eating habits, changes in health do not manifest instantly, rather we could expect either a slow increase or slow reduction, depending on whether the change of habits was positive or negative. Thus, for many measurements, the change in health from the previous measurement will likely be in the same direction as the change was one step earlier. The following details how to specify such a model, generate data using the `ctGenerate` function, simply plot the generated data, and estimate the parameters.

```
genm <- ctModel(Tpoints = 200, n.latent = 2, n.manifest = 1,
  LAMBDA = matrix(c(1, 0), nrow = 1, ncol = 2),
  DIFFUSION = matrix(c(0, 0, 0, 1), 2),
  MANIFESTVAR = t(chol(diag(.6,1))),
  DRIFT = matrix(c(0, -.1, 1, -.2), nrow = 2),
  CINT = matrix(c(1, 0), nrow = 2))

data <- ctGenerate(genm, n.subjects = 1, burnin = 200)

ctIndplot(data, n.subjects = 1 , n.manifest = 1, Tpoints = 200)

model <- ctModel(Tpoints = 200, n.latent = 2, n.manifest = 1,
  LAMBDA = matrix(c(1, 0), nrow = 1, ncol = 2),
  DIFFUSION = matrix(c(0, 0, 0, "diffusion"), 2),
  DRIFT = matrix(c(0, "regulation", 1, "diffusionAR"), nrow = 2))

fit <- ctFit(data, model, stationary = c("T0VAR"))
```

In the above, we focus on a model for a single subject, and specify with LAMBDA that a single manifest variable measures only the first latent process. With DIFFUSION we specify that only the 2nd unobserved process experiences random innovations. With DRIFT, we specify that the 2nd process has a freely estimated autoregression term, that it directly impacts the first process with a 1:1 relationship, and that as the level of the first process increases, the level of the 2nd process decreases – providing necessary regulation.

Damped linear oscillator

Voelkle and Oud (2013) discuss modelling a damped linear oscillator in detail, however here we demonstrate how to load the data and fit the oscillating model from their paper. In this case, we also specify good starting values with the `startValues` argument to `ctModel`, and because of this, set the argument `carefulFit = FALSE`.

```
data("Oscillating")

inits <- c(-39, -.3, 1.01, 10.01, .1, 10.01, 0.05, .9, 0)
names(inits) <- c("crosseffect","autoeffect", "diffusion",
  "T0var11", "T0var21", "T0var22","m1", "m2", "manifestmean")
```

```
oscillatingm <- ctModel(n.latent = 2, n.manifest = 1, Tpoints = 11,
  MANIFESTVAR = matrix(c(0), nrow = 1, ncol = 1),
  LAMBDA = matrix(c(1, 0), nrow = 1, ncol = 2),
  TOMEANS = matrix(c("m1", "m2"), nrow = 2, ncol = 1),
  TOVAR = matrix(c("T0var11", "T0var21", 0, "T0var22"), nrow = 2, ncol = 2),
  DRIFT = matrix(c(0, "crosseffect", 1, "autoeffect"), nrow = 2, ncol = 2),
  MANIFESTMEANS = matrix("manifestmean", nrow = 1, ncol = 1),
  DIFFUSION = matrix(c(0, 0, 0, "diffusion"), nrow = 2, ncol = 2),
 startValues=inits)

oscillatingf <- ctFit(Oscillating, oscillatingm, carefulFit = FALSE)
```

2.7. Additional specification options and tips for model estimation

Given the complexity of parameter constraints, and that for some classes of models multiple minima may exist, parameter estimation is sometimes difficult. To ensure reliable estimation, there are some additional approaches that may be helpful. By default the argument `carefulFit = TRUE` for the `ctFit` function is specified. This initiates a two-step procedure, in which the first step penalises the likelihood[7] to help maintain potentially problematic parameters close (but not too close!) to 0, and then uses these estimates as starting values for maximum likelihood estimation. Though often useful, in some cases (particularly those with complex dynamics, or user specified starting values) it can help to switch this off. Beyond this, as a general guideline we suggest starting with simpler, more constrained models and freeing parameters in a stepwise fashion (not necessary as a means of model development, simply for fitting purposes). The `ctRefineTo` function can be used as a replacement for `ctFit` and automates this step-wise progression. One could do this manually by developing the measurement model separately, estimating only autoregressive parameters of the DRIFT matrix at first (in simple models, this means constraining the off-diagonals of the DRIFT matrix to 0), or fixing the factor loading matrix prior to free estimation. For such stepwise progression, the default of small and negative starting values for cross effects should be switched off by argument `crossEffectNegStarts = FALSE`, and starting values from the restricted model should be obtained via the `omxStartValues` argument, as shown here, using the model fit from from Section 2.5:

```
omxInits <- omxGetParameters(example1fit$mxobj)

fitWithInits <- ctFit(data = ctExample1, ctmodelobj = example1model,
  omxStartValues = omxInits)
```

If stepwise model building with starting values based on simpler fits still fails to produce an improved solution, some of the following suggestions may be helpful. The time scale, although theoretically unimportant in the sense that all time ranges can be accounted for, can be computationally relevant. It is helpful to choose a scale that roughly matches the expected dynamics – for instance a time scale of nanoseconds for panel data measured yearly would be problematic, instead, a yearly or monthly time scale could be used. Centering the *grand* mean of the variables to 0 may help, as can standardising the variances, particularly in cases where both a measurement model and dynamic model are estimated. One way to search for an improved solution is simply to try many times with varying starting values. This is automated by default

[7] The sum of squares of each parameter that is neither factor loading nor mean related is added to the likelihood, as is the inverse of any parameters on the diagonals of square matrices – essentially penalising values at the extremes

using the `mxTryHard` function from OpenMx, however you may want to increase the `retryattempts` argument to `ctFit`, or simply re-run `ctFit` many times, as it generates unspecified starting values with some limited randomness. However, since both automated procedures begin within a similar range, for truly problematic cases one may consider adding more extensive randomness to the starting values manually. In situations with a limited number of time points or of high complexity, you may implement the stationarity assumption, so that parameters related to the first time point are no longer estimated, but constrained to the asymptotic effects, when the time interval $\Delta t \to \infty$. This can make optimization more straightforward, and may serve as a useful basis for determining starting values, or as a viable model in itself. For more discussion regarding stationarity conditions see Section 2.3.3. In some cases, optimizing over the continuous time parameters (the default) results in fits that do not pass every check and you may be left with warnings. If this occurs, you can instead optimize over asymptotic variants of the parameters, by using the argument `asymptotes = TRUE` to `ctFit`. This will in general produce equivalent results if the DIFFUSION, CINT, and TIPRED matrices are all freely estimated, even though the raw parameters of these matrices will look different. If these matrices have been constrained in some way, this approach is not recommended.

2.7.1. Optimization performance

When time intervals vary for every individual, optimization can be quite slow. To quickly estimate approximate versions of a model, you may use the `meanIntervals = TRUE` argument to `ctFit`, which will set every individual's time intervals to the mean of the interval across all individuals. A step further even is to specify the argument `objective = "cov"` in order to estimate a covariance matrix from the supplied data and fit directly to that. In cases with variability in time intervals these approaches will substantially speed up optimization, but also waste information and bias parameters. Using such an approach in combination with a constrained DRIFT matrix to generate starting values may be a reasonable way to improve fitting speed and generate starting values for large and complex models.

2.8. Chapter summary

While there are some difficulties and limitations to the frequentist SEM approach, such as difficult optimization when the data are not sufficiently informative, and limitations with respect to individual differences, the ctsem software as it stands allows for the straightforward specification of continuous time structural equation models for both panel and time series data. The models may include multiple, potentially higher order processes, multiple indicators with a measurement model, exogenous predictors, unobserved heterogeneity, multiple groups, as well as constraints between different parameters. We hope that the software assists the understanding and analysis of dynamic models with panel data.

Chapter 3.

Hierarchical Bayesian Continuous Time Dynamic Models

In this work we bring continuous time dynamic modeling together with hierarchical Bayesian modeling, and describe the resultant model and software for estimation. In this approach, the estimation of subject specific continuous time dynamic model parameters is enhanced by using data from other subjects to inform a prior distribution, with no restrictions on the length of time series or number of subjects. This allows for questions regarding individual differences in dynamics, without the requirement of very large numbers of repeated observations that single-subject time series approaches can have, nor the convergence problems with low numbers of subjects that some random-effects approaches have.

Both hierarchical Bayes, and continuous time dynamic modeling, offer an improved use of data over common alternatives. The hierarchical approach uses information from other subjects to better estimate parameters of each subjects‘ specific model, while the continuous time approach uses information about the time interval between measurements of each subject. The nature of this improvement is such that it also allows for new questions on individual differences in the continuous time model parameters. That is, in datasets that do not contain sufficient information for single-subject time series approaches – as for instance with the German socieconomic panel (GSOEP) – we can now estimate subject-specific continuous time dynamic models, by incorporating information from other subjects. In addition to these improvements, the continuous time approach allows for the parsimonious description of higher order systems involving oscillations and multiple time-scales. We first then consider the combination of both approaches, then move on to describe the full model more formally. Following this, we discuss the software implementation we have developed, demonstrate performance of the approach and software with a simulation study, and finish with an example application of a dynamic model of wellbeing based on the GSOEP data. Although the article aims at creating a general understanding of the model, readers who are primarily interested in its application, may easily skip over some of the more detailed description of the subject likelihood and population distribution in the model section.

3.0.1. Hierarchical continuous time dynamic models

While Oud and Jansen (2000) describes a mixed-effects, structural equation modelling approach to continuous time dynamic models, and Driver et al. (2017) the related software, relatively little work has been done on the generalization to fully random-effects models, in which all parameters may vary over subjects. The continuous time dynamic model has been specified in Bayesian form by Boulton (2014), but this did not extend to a hierarchical approach. The most substantial foray into hierarchical continuous time approaches in psychology has been with the work by Oravecz, Tuerlinckx and Vandekerckhove (2009, 2016) on the hierarchical Ornstein-Uhlenbeck model, which also led to the creation of the BHOUM software for estimation. While the software and modelling approach as it stands is useful, there is a limitation for some

applications in that the matrix containing dynamic effects is constrained to be symmetric and positive definite. This means that common model structures such as the autoregressive and cross lagged model, or damped linear oscillator, cannot be estimated, because cross-effects from a first process to a second take the same value as cross-effects from that second process back to the first.

Chow, Lu, Sherwood and Zhu (2014) have also looked at hierarchical dynamics in continuous time, fitting nonlinear ordinary differential equation models with random effects on the parameters to ambulatory cardiovascular data from multiple subjects. While the nonlinear aspect offers increased flexibility (at cost of additional complexity), the ordinary differential aspect, rather than stochastic differential, means that the estimated processes are deterministic, and randomness is assumed to occur only at the measurement level. Unless the process is very well understood, with all contributing factors measured, the lack of innovation variance may be quite limiting. (Lu, Chow, Sherwood & Zhu, 2015) used similar estimation algorithms and data with stochastic differential equations which can account for innovation variance, but avoided a hierarchical approach, and assumed independence between subjects parameters.

This work describes the model and software for a hierarchical Bayesian continuous time dynamic model, without any restriction of symmetric positive-definite dynamics. It builds upon the model and software, ctsem, described in Driver et al. (2017), which offered a mixed-effects modeling framework for the same linear stochastic differential equation model, in which intercept and mean related parameters could be estimated as random effects across subjects, but subject level variance and regression parameters were restricted to fixed effects estimation. With the change to a hierarchical Bayesian framework, all subject level parameters may now vary across subjects, the covariance between subject level parameters is available, and the estimation of time-invariant relations between covariates and all subject level parameters is now possible (as opposed to relations with only the mean levels of subjects processes).

3.1. The model

There are three main elements to our hierarchical continuous time dynamic model. There is a subject level latent dynamic model, a subject level measurement model, and a population level model that describes the distribution of subject level parameters across subjects. Note that while various elements in the model depend on time, the fundamental parameters of the model are time-invariant. Note also that we ignore subject specific subscripts when discussing the subject level model. The subject level dynamics are already described in Equation 1.1, and the measurement model in Equation 1.9, which are covered in detail in Chapter 1, Pages 7 and 10. For ease of reference, the two equations are repeated here:

$$\mathrm{d}\boldsymbol{\eta}(t) = \Big(\mathrm{A}\boldsymbol{\eta}(t) + \mathrm{b} + \mathrm{M}\boldsymbol{\chi}(t)\Big)\mathrm{d}t + \mathrm{G}\mathrm{d}\mathrm{W}(t) \tag{3.1}$$

$$\mathrm{y}(t) = \Lambda\boldsymbol{\eta}(t) + \boldsymbol{\tau} + \boldsymbol{\epsilon}(t) \quad \text{where } \boldsymbol{\epsilon}(t) \sim \mathrm{N}(0_c, \Theta) \tag{3.2}$$

3.1.1. Subject level likelihood

The subject level likelihood, conditional on time dependent predictors x and subject level parameters Φ, is as follows:

$$p(\mathrm{y}|\Phi, \mathrm{x}) = \prod^{\mathrm{U}} p(\mathrm{y}_u|\mathrm{y}_{\left(u-1,\ldots,u-(u-1)\right)}, \mathrm{x}_u, \Phi) \tag{3.3}$$

To avoid the large increase in parameters that comes with sampling or optimizing latent states, we use a continuous-discrete, or hybrid, Kalman filter (Kalman & Bucy, 1961) to ana-

lytically compute subject level likelihoods, conditional on subject parameters. For more on filtering see Jazwinski (2007) and Särkkä (2013). The filter operates with a prediction step, in which the expectation $\hat{\boldsymbol{\eta}}_{u|u-1}$ and covariance $\hat{\mathrm{P}}_{u|u-1}$ of the latent states are predicted by:

$$\hat{\boldsymbol{\eta}}_{u|u-1} = \mathrm{A}^*_{\Delta t_u}\hat{\boldsymbol{\eta}}_{u-1|u-1} + \mathrm{b}^*_{\Delta t_u} + \mathrm{M}\mathrm{x}_u \tag{3.4}$$

$$\hat{\mathrm{P}}_{u|u-1} = \mathrm{A}^*_{\Delta t_u}\hat{\mathrm{P}}_{u-1|u-1}(\mathrm{A}^*_{\Delta t_u})^\top + \mathrm{Q}^*_{\Delta t_u} \tag{3.5}$$

For the first measurement occasion $u = 1$, the values $\hat{\boldsymbol{\eta}}_{u|u-1}$ and $\hat{\mathrm{P}}_{u|u-1}$ must be provided to the filter. These parameters may in some cases be freely estimated, but in other cases need to be fixed or constrained, either to specific values or by enforcing a dependency to other parameters in the model, such as an assumption of stationarity.

Prediction steps are followed by an update step, wherein rows and columns of matrices are filtered as necessary depending on missingness of the measurements y. The update step involves combining the observed data with the expectation and variance, using the Kalman gain matrix $\mathrm{K} \in \mathbb{R}^{v\times c}$, which represents the ratio between the process inovation covariance and measurement error.

$$\hat{\mathrm{y}}_{u|u-1} = \Lambda\hat{\boldsymbol{\eta}}_{u|u-1} + \boldsymbol{\tau} \tag{3.6}$$

$$\hat{\mathrm{V}}_u = \Lambda\hat{\mathrm{P}}_{u|u-1}\Lambda^\top + \Theta \tag{3.7}$$

$$\hat{\mathrm{K}}_u = \hat{\mathrm{P}}_{u|u-1}\Lambda^\top\hat{\mathrm{V}}_u^{-1} \tag{3.8}$$

$$\hat{\boldsymbol{\eta}}_{u|u} = \hat{\boldsymbol{\eta}}_{u|u-1} + \hat{\mathrm{K}}_u(\mathrm{y}_u - \hat{\mathrm{y}}_{u|u-1}) \tag{3.9}$$

$$\hat{\mathrm{P}}_{u|u} = (\mathrm{I} - \hat{\mathrm{K}}_u\Lambda)\hat{\mathrm{P}}_{u|u-1} \tag{3.10}$$

The log likelihood (ll) for each subject, conditional on subject level parameters, is typically[1] then (Genz & Bretz, 2009):

$$\begin{aligned} ll = \sum^{\mathrm{U}}\Bigg(&- 1/2(n\ln(2\pi) + \ln\left|\mathrm{V}_u\right| + \\ &(\hat{\mathrm{y}}_{(u|u-1)} - \mathrm{y}_u)\mathrm{V}_u^{-1}(\hat{\mathrm{y}}_{u|u-1} - \mathrm{y}_u)^\top)\Bigg) \end{aligned} \tag{3.12}$$

Where n is the number of non-missing observations at measurement occasion u.

[1] For computational reasons we use an alternate but equivalent form of the log likelihood. We scale the prediction errors across all variables to a standard normal distribution, drop constant terms, calculate the log likelihood of the transformed prediction error vector, and appropriately update the log likelihood for the change in scale, as follows:

$$ll = \sum^{\mathrm{U}}\left(ln\big(tr(\mathrm{V}_u^{-1/2})\big) + \sum 1/2\big(\mathrm{V}_u^{-1/2}(\hat{\mathrm{y}}_{(u|u-1)} - \mathrm{y}_u)\big)\right) \tag{3.11}$$

Where tr indicates the trace of a matrix, and $\mathrm{V}^{-1/2}$ is the inverse of the Cholesky decomposition of V. The Stan software manual discusses such a *change of variables* (Stan Development Team, 2016b).

3.1.2. Population distribution

Rather than assume complete independence or dependence across subjects, we assume subject level parameters are drawn from a population distribution, for which we also estimate parameters, conditional on specified hyperpriors. This results in a joint-posterior distribution of:

$$p(\Phi, \boldsymbol{\mu}, \mathrm{R}, \boldsymbol{\beta} | \mathrm{Y}, \mathrm{Z}) = \frac{p(\mathrm{Y}|\Phi)p(\Phi|\boldsymbol{\mu}, \mathrm{R}, \boldsymbol{\beta}, \mathrm{Z})p(\boldsymbol{\mu}, \mathrm{R}, \beta)}{p(\mathrm{Y})} \tag{3.13}$$

Where subject specific parameters Φ_i are determined in the following manner:

$$\Phi_i = \mathrm{tform}\Big(\boldsymbol{\mu} + \mathrm{R}\mathrm{h}_i + \boldsymbol{\beta}\mathrm{z}_i\Big) \tag{3.14}$$

$$\mathrm{h}_i \sim \mathrm{N}(\mathbf{0}, 1) \tag{3.15}$$

$$\boldsymbol{\mu} \sim \mathrm{N}(\mathbf{0}, 1) \tag{3.16}$$

$$\boldsymbol{\beta} \sim \mathrm{N}(\mathbf{0}, 1) \tag{3.17}$$

The basic structure of Equation 3.14 is such that everything inside the brackets – population distribution means, subject specific random deviations, and covariate effects – is on the unconstrained, real number scale. This bracketed portion, which we will later refer to as the *raw* subject level parameters, then undergoes some transformation that varies depending on the sort of parameter to be estimated (e.g., standard deviations need to be positive, correlations must be between -1 and 1). This transformation of the raw subject level parameters then leaves us with the subject level parameters we are actually interested in. For this reason, we will also refer to the population means and covariate effects inside the brackets as raw population means and raw covariate effects – they are a necessity but we are not interested in them directly. We take this approach to ensure that subject specific parameters do not violate boundary conditions, and that deviations from a mean that is close to a boundary are more likely to be smaller in the direction of the boundary than away from it. For example, for a standard deviation parameter with a population distribution mean of 0.30, a subject specific deviation of -0.40 is not possible because it results in a negative value, while $+0.40$ would be perfectly reasonable. Figure 3.1 gives a visual sense to this approach with transformations. Regarding Equation 3.14 more specifically: $\Phi_i \in \mathbb{R}^s$ is the s length vector of parameters for the dynamic and measurement models of subject i. $\boldsymbol{\mu} \in \mathbb{R}^s$ parameterizes the means of the raw population distributions of subject level parameters. $\mathrm{R} \in \mathbb{R}^{s \times s}$ is the Cholesky factor of the raw population distribution covariance matrix, parameterizing the effect of subject specific deviations $\mathrm{h}_i \in \mathbb{R}^s$ on Φ_i. The Cholesky factor R is itself a transformation of parameters sampled and transformed in various ways, as discussed in the following section. $\boldsymbol{\beta} \in \mathbb{R}^{s \times w}$ is the raw effect of time independent predictors $\mathrm{z}_i \in \mathbb{R}^w$ on Φ_i, where w is the number of time independent predictors. Y_i contains all the data for subject i used in the subject level model – y (process related measurements) and x (time dependent predictors). z_i contains time independent predictors data for subject i. tform is an operator that applies a transform to each value of the vector it is applied to. The specific transform depends on which subject level parameter matrix the value belongs to, and the position in that matrix — these transforms and rationale are described below, but are in general necessary because many parameters require some bounded distribution, making a purely linear gaussian approach untenable.

The approach wherein we first sample raw subject specific deviations h_i from a standard normal distribution, before multiplying them by the Cholesky factor of the population distribution R, may be unfamiliar to some. This is a non-centered parameterization, which we implemented to improve sampling efficiency. See Bernardo et al. (2003) and Betancourt and Girolami (2013) for discussion of non-centered parameterizations.

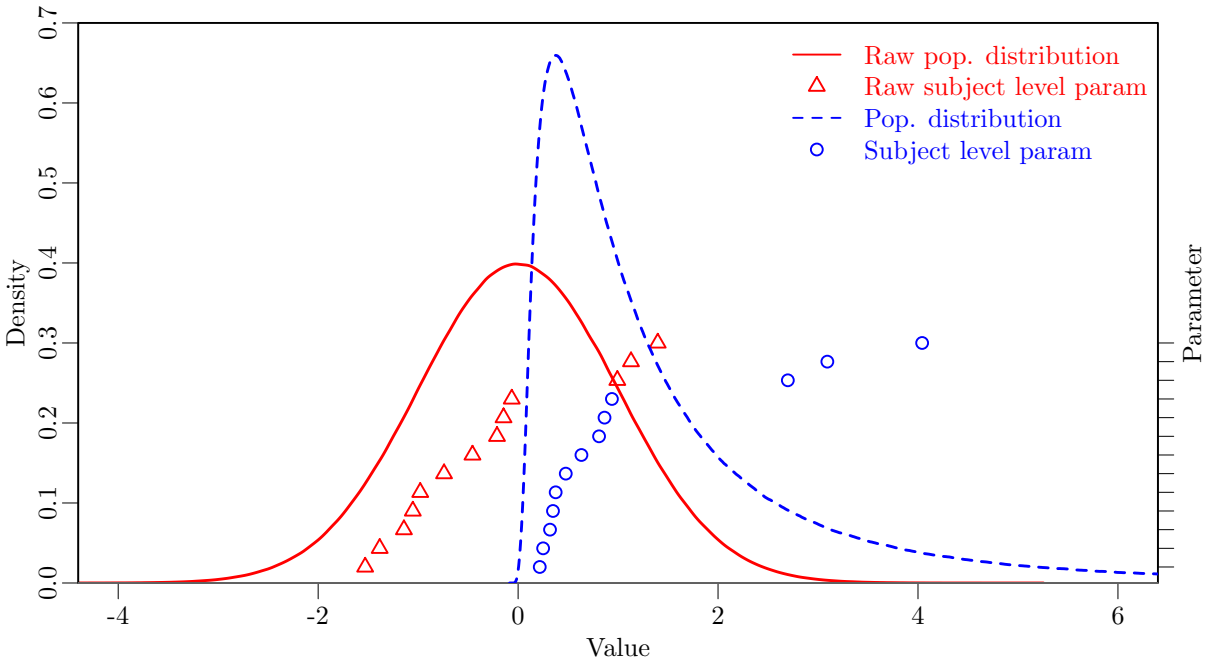


Figure 3.1.: Subject level parameters are obtained by first sampling from a multivariate normal *raw* population distribution, and then transforming by some function to satisfy boundary and other criteria. This example shows a possibility for standard deviation parameters, using an exponential transformation. The height of the individual parameters simply represents the match between raw and transformed.

3.1.3. Cholesky factor of population distribution covariance

The matrix R, the Cholesky factor [2] of the population distribution covariance matrix, accounts for parameter correlations such as would be found when, for example, subjects that typically score highly on measurements of one process are also likely to exhibit stronger auto-effects on another. Rather than simply using the conjugate inverse-Wishart prior for the covariance matrix, we opt for an approach that first separates scale and correlation parameters (Barnard, McCulloch & Meng, 2000), as shown in Equation 3.18

$$\mathrm{Z} = \mathrm{RR}^{\top} \tag{3.18}$$

$$\mathrm{R} = \mathrm{SX} \tag{3.19}$$

Z is a covariance matrix, R is the Cholesky factor of covariance Z, X is a Cholesky factor correlation matrix, and S is a diagonal matrix of standard deviations. These matrices are all of dimension $s \times s$. This separation approach is taken to ensure that the scale of population distributions does not influence the prior, and posterior, probability of the correlation between them (Tokuda, Goodrich, Van Mechelen, Gelman & Tuerlinckx, 2011), and to ensure that the prior distribution does not become highly informative as variances approach zero (Gelman, 2006). However, a difficulty still remains in that there is a position dependency introduced through the use of Cholesky factors, as values in higher columns depend on those in the lower, but not vice versa. To account for this, we sample the Cholesky factor correlation matrix using an *LKJ* prior. This is implemented directly in the Stan software and recommended in

[2] For those unfamiliar, the Cholesky factor of a symmetric positive definite matrix is simply a lower or upper triangular matrix that, when multiplied by the transpose of itself, gives the original symmetric positive definite matrix.

the user guide (Stan Development Team, 2016a), and is based on the work of Lewandowski, Kurowicka and Joe (2009). Specifics of how the prior is obtained are complex, but the result is that of a uniform distribution over the space of correlation matrices, and prior probabilities of correlations are consistent across all elements of the matrix.

To obtain the raw population distribution Cholesky factor covariance matrix R, the Cholesky factor correlation matrix X is multiplied by a diagonal matrix S containing standard deviations, where the standard deviations are sampled with a standard normal prior distribution, truncated below zero. Such a standard normal prior for the raw population distribution works, because the expected scale of subject level parameters is set by the tform function, described in the following section. When necessary, the prior for the scale of raw population distributions can be changed from this standard normal, by multiplying the vector of standard deviations by a scaling vector that is fixed in advance. The result of all this is simply that we obtain an efficient implementation of the Cholesky factor covariance matrix of the raw population distribution, which gives a multivariate normal prior distribution with a scale of 1 and uniform prior over the space of correlation matrices.

3.1.4. tform operator - parameter transforms and priors

The tform operator achieves two things. The first, is to convert the raw subject level parameters of each subjects dynamic and measurement models, from the standard normal space we use for sampling, to a range of differently shaped distributions. The second, is to set the prior distribution over our parameter space. Because the raw parameters are on a standard normal scale, applying a simple linear transform multiplying by two, would result in a prior with a standard deviation of 2, for the parameter of interest. In general, the transformations and resulting priors we discuss here are proposed as reasonable starting points for a range of typical situations, not as perfectly robust catch-all solutions. The ctsem software we discuss later allows for them to be easily altered. The transforms we choose, and resulting shape of the prior distributions, depends on the requirements of the specific parameter types. For instance as we have already discussed, standard deviation parameters cannot be negative, so a simple approach for those is for the tform operator to perform an exponential operation. Such parameter boundaries also imply that the subject level parameters are unlikely to be normally, or symmetrically, distributed, particularly as means of the population distributions approach the boundaries – a change in standard deviation from 1.00 to 0.01 is more dramatic than a change from 1.00 to 1.99. Along with the general shape and boundaries of the prior distribution, the scale is of course also important. As a general approach we have aimed for scales that support inference using standardised and centered data, with parameters at relatively normal magnitudes – neither extremely large nor extremely small, as such parameters will tend to generate numeric problems anyway. Further, when subject level parameters are estimated at an upper or lower bound in such a model, it can indicate the need for model respecification (such as including higher order terms) or rescaling the time variable. Another factor to take into account with regard to transformation, is that there is also a need to be able to fix parameters to specific, understandable values, as for instance with elements of the diffusion matrix Q, which for higher order models will generally require a number of elements fixed to 0. This possibility can be lost under certain multivariate transformations. A final important factor for deciding on a transformation is that of sampling efficiency. Sampling efficiency is typically reduced when parameters are correlated (with respect to the sampling procedure), because a random change in one parameter requires a corresponding non-random change in another to compensate, complicating efficient exploration of the parameter space. While the use of modern sampling approaches like Hamiltonian Monte Carlo (Betancourt & Girolami, 2013) and the no U-turn sampler (Homan & Gelman, 2014) mitigate these issues to some extent, minimizing correlations between parameters through transformations still substantially improves

performance. A further efficiency consideration is the inclusion of sufficient prior information to guide the sampler away from regions where the likelihood of the data approaches zero and the gradient of the likelihood is relatively flat.

Covariance matrices are created in a similar way to that described for the population distribution Cholesky factor covariance, in that we multiply a Cholesky factor correlation matrix by a diagonal matrix containing the standard deviations to obtain a Cholesky factor covariance, and then multiply this matrix by its transpose to obtain a full covariance matrix. There are some additional complexities with respect to the Cholesky factor correlation however. To begin, we obtain the partial correlation values for the lower-triangle of the Cholesky factor correlation matrix, by transforming raw parameters to the range -1 to 1, via a scaled inverse logit function.

$$\Phi_{ij} = 2/(1 + e^{-x}) - 1 \tag{3.20}$$

Where x is the raw parameter and j denotes the location of any partial correlation parameters in the subject level parameter vector Φ_i. Because we need to be able to fix values in advance, the built in Stan functionality for the LKJ prior is not possible, so we implement the 'algorithm for generating correlation matrices with vines' from Lewandowski et al. (2009). Normally, this updates the log probabilities of the partial correlation parameters depending on their location in the matrix, and calculates the appropriate diagonal for the matrix. However, because we only want the LKJ prior on the *population mean distribution* of the Cholesky factor correlation, but need the subject specific Cholesky factor correlations that result from combining the population mean with subject specific deviations, we separate the algorithm into two parts. For the first part, we update the log probabilities of the population distribution means of the partial correlation parameters, depending on which element of the matrix they belong to. For the second part, we calculate the Cholesky factor correlation matrix for each subject as per the algorithm, but neglecting the log probability update. An R and Stan script showing this approach to handling hierarchical correlation matrices is provided in Appendix E. The script includes a simulation routine to plot resulting prior densities of correlations, for a specified dimension and standard deviation. Based on our own testing, results in general look very good, though when there is high variance in the population distribution and higher dimensional correlation matrices, some very mild differences of prior distributions of correlations occur, dependent on which indices of the matrix the correlation is for. The field of random correlation matrices and hierarchical correlation matrices is in general still developing, we are not aware of a perfect solution at this point.

The subject specific Cholesky factor correlation matrices (of, for example, the continuous time DIFFUSION matrix) are multiplied by a diagonal matrix containing standard deviations, to obtain a Cholesky factor covariance, which is then multiplied by its own transpose to obtain a covariance matrix. Standard deviation parameters within the subject level models are obtained by exponentiating a multiple of the raw parameter, which results in a prior similar to the Jeffreys or reference scale prior (Bernardo, 1979), but is regularized away from the low or high extremes to ensure a proper posterior. This form, wherein mass reduces to 0 at any boundaries, is used for all subject level parameters where boundaries exist, because typically at such boundaries other parameters of the model become empirically non-identified and optimization or sampling procedures can run into trouble.

$$\Phi_{ij} = e^{4x} \tag{3.21}$$

Where x is the raw parameter and j denotes the location of any partial correlation parameters in the subject level parameter vector Φ_i.

Because intercept and regression type parameters need not be bounded, for these we simply scale the standard normal to the desired range (i.e., level of informativeness) by multiplication. We could of course also add some value if we wanted a non-zero mean.

Diagonals of the drift matrix A – the temporal auto effects – are transformed to be negative, with probability mass relatively uniformly distributed for discrete time autoregressive effects between 0 and 1, given a time interval of 1, but declining to 0 at the extremes. We have opted to use a bounded distribution on the drift auto effects for pragmatic reasons – values greater than 0 represent explosive, non-stationary processes that are in most cases not theoretically plausible and occur only due to model misspecification. Further, positive values can result in additional local minima, causing problems for optimization or sampling. While allowing for such values may point to misspecification more readily, the constrained form results in what we believe is a computationally simpler and sensible prior distribution for genuine effects – but the model and software allows for this to be easily changed.

$$\Phi_{ij} = -log(e^{-1.5x} + 1) \tag{3.22}$$

Where x is the raw parameter and j denotes the location of any partial correlation parameters in the subject level parameter vector Φ_i.

We have found that off diagonals of the drift matrix A – the temporal cross effects – function best when specified in a problem dependent manner. For simple first order processes, they can simply be left as multiplications of the standard normal distribution. For higher order processes, the cross effects between a lower and higher order component, determining for instance the wavelength of an oscillation, may need to be parameterized similarly to the auto effects, ensuring negative values. This is so that optimization or sampling does not get stuck at a local minimum just above zero.

Figure 3.2 plots the resulting prior densities when using the described transformations. Note that of course the density for a variance is directly related to the standard deviation, and the density plot for an autoregression assumes that the time interval is 1 with no cross effects involved. For the sake of completeness we include a prior density for all other parameters, such as the drift cross effects, intercepts, and regression type parameters, although these just use a simple multiplication of the standard normal.

3.2. Software implementation

The hierarchical continuous time dynamic model has been implemented as an extension to the ctsem software (Driver et al., 2017) for R (R Core Team, 2014). Originally, ctsem was designed to perform maximum likelihood estimation of continuous time structural equation models as they are described in Voelkle et al., 2012, in which the structural equation matrices are set up in the RAM (reticular action model) format (J. Jack McArdle & McDonald, 1984). Individual specific time intervals are accounted for by *definition variables*, and these are coupled with matrix algebra functions to determine the expected means and covariance matrices for each individual. The need for complex functions like the matrix exponential made the OpenMx software (Neale et al., 2016) an obvious choice for fitting the models. In this original form of ctsem however, random-effects are only possible to estimate for intercept parameters. This is a primary motivation for this extension to a hierarchical Bayesian formulation, where all parameters may vary across individuals according to a simultaneously estimated distribution. To fit this new hierarchical form of the model, we use a recursive state-space formulation in which expectations for each time point are modeled conditional on the prior time point, and rely on the Stan software (Carpenter et al., 2017) for model estimation and inference.

Stan is a probabilistic programming language with some similarities to the BUGS language (Bayesian inference using Gibbs sampling) (Spiegelhalter, Thomas, Best, Gilks & Lunn, 1996) language, but greater flexibility. While the switch to a hierarchical Bayesian approach offers a range of benefits, it comes at the price of additional computation time, and the necessity

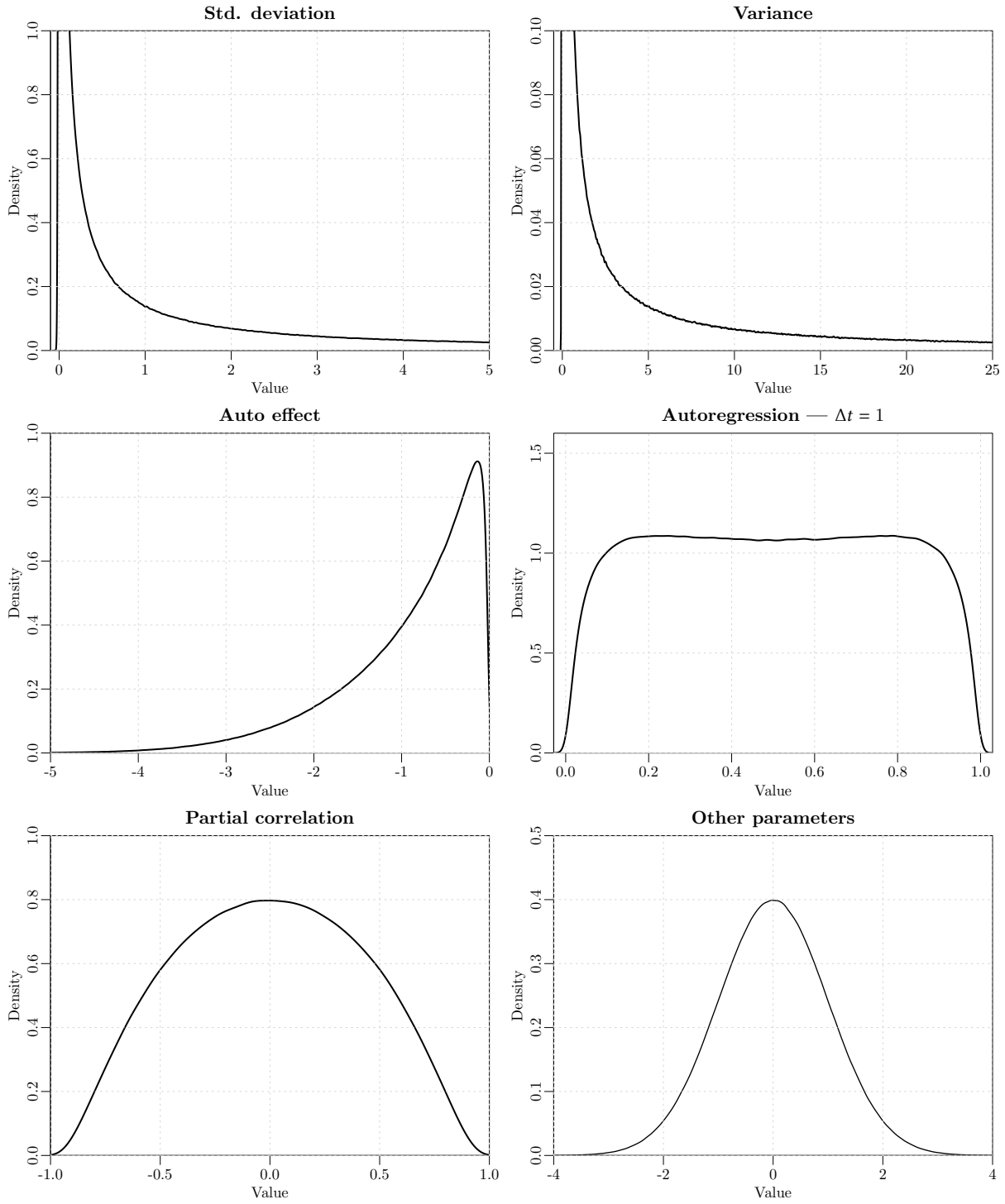


Figure 3.2.: Priors for population distribution means.

of specifying priors. In cases where computation time or the use of priors is problematic, or one wishes to develop a specific model structure not available with a recursive state-space formulation, the classic form of ctsem for frequentist inference may be used via some different function arguments.

3.2.1. Software usage

The ctsem (Driver et al., 2017) software is available via the R software repository CRAN, using the R code `install.packages('ctsem')`. While full details on usage of the software are provided in the help files of the mentioned functions and the supplementary ctsem package vignette, 'Introduction to Hierarchical Continuous Time Dynamic Models With ctsem', http://cran.r-project.org/package=ctsem/vignettes/hierarchical.pdf, we describe the fundamentals here. The main functions of this extension to ctsem are the ctModel and ctStanFit functions. The ctModel function allows the user to specify the continuous time matrices as any combination of fixed values or freely estimated parameters. The ctStanFit function translates the specification from ctModel into a model in the Stan language, combines this model with specified data, and estimates the model. Summary and plot functions are available for the output object, and additional details are available by directly applying rstan (Stan Development Team, 2016a) functions to the rstan fit output, available within the ctStanFit output as myfit$stanfit.

3.3. Simulation study

To confirm and demonstrate the performance of our specification of the hierarchical Bayesian continuous time dynamic model, we have conducted a small simulation study. For the study, we used a model similar to that in the empirical study of wellbeing dynamics we show in the next section, with true parameter values of the simulation similar to those estimated by the empirical work. In order to contrast the Bayesian implementation with a frequentist form, the generating model we used is a mixed-effects model, with individually varying intercepts but dynamics fixed across subjects. Using the ctsem software, we fit data from this generating model using the true model with both Bayes and frequentist forms, and also fit using the fully random-effects Bayesian model, in order to provide guidance regarding performance when the population model may be over specified. We generated data for either 10 or 30 observation time points, 50 or 200 subjects, a cross effect of 0.00 or 0.20, and with or without variation in time intervals between observations. The conditions were fully crossed, and each condition was repeated 200 times. To ensure that only results from converging and reasonable fits were used for inference, we excluded runs if the OpenMx optimizer reported a problem, if the frequentist point estimates were so extreme as to be clearly ridiculous, or if any split Rhat scale reduction factor Gelman and Rubin, 1992 from the Bayesian samples was greater than 1.10, or the number of effective samples for any parameter was lower than 50 (50 would generally be regarded as too low for inference regarding 95% intervals, but is nevertheless informative in the aggregate as here). The split Rhat value refers to the ratio of between chain to within chain variance, while the number of effective samples calculate the effective number of samples out of the total drawn, after accounting for autocorrelation in the samples. Only roughly 1% of results were removed due to issues on the frequentist side, but there were more due to Bayesian issues. This is not due to inherent difficulties, but simply that in order to maximise the number of simulation runs we could make, the number of Bayesian iterations was set to 300 warmup and 300 sampling, resulting in some runs that did not yet meet criteria (Note that the Hamiltonian MCMC used in Stan is much more efficient per iteration than many other sampling approaches, but also much slower per iteration). One limitation of the implementation using Stan is that when convergence criteria are not reached, there is no easy way to continue sampling – one

must restart from the beginning. Overall, 16% of runs were rejected due to inadequate effective samples, while very few (< 1%) that passed this criterion were rejected due to an inadequate scale reduction factor on one or more variables. Conditions with fewer time points and subjects, and fit with full random effects had in general more rejected runs – here increasing the number of iterations may have been more efficient. We did not modify the priors for the Bayesian approach from those discussed above, and used only default starting values given by the ctsem software

Tables 3.1 and 3.2 show coverage rates, true values and mean point estimates, and RMSE (root mean squared error), for all estimated parameters, along with the symbol representing the parameter in the subject level model (if the parameter is a subject level parameter). These are shown for the Bayesian mixed-effects model (B true), Bayesian fully random effects model (B full), and frequentist model, with either 200 subjects measured at 30 time points in Table 3.1, or 50 subjects and 10 time points in Table 3.2. For frequentist intervals, we used both a profile likelihood approach and a Wald type approach based on the standard error. The profile likelihood approach is slower, not based on asymptotic theory, non-symmetric around the optimized parameter, and is generally regarded as offering improved inferential properties (Evans, Kim & O'Brien, 1996). OpenMx offers a range of optimizer choices, which we have observed impact confidence interval estimation – based on past experience we are using the NPSOL (Gill, Murray, Saunders & Wright, 1986) optimizer. The Wald type CI approach is a simple calculation based on the Hessian, but is strictly valid only as the sample tends towards infinity and may be in particular problematic for bounded parameters. Due to limitations with the number of simulation runs possible, the simulation measures reported will be subject to some sampling variability, and thus should not be treated as perfectly precise – nevertheless they offer insight into the positives and negatives of the discussed Bayesian approach.

Table 3.1 shows that with 200 subjects and 30 time points, inferential properties are for the most part very good regardless of the approach taken – empirical coverage rates of the 95% intervals are approximately 95%, there is minimal to no bias in parameters, and error from point estimates is comparable across approaches. The T0VAR parameters, representing the initial covariance matrix of the latent processes, are an exception to this, though these are in general nuisance parameters required for initialisation and not interesting for inference, and are estimated based on quite limited information (in a single subject approach they must also generally be fixed in advance, in any case). Correlation parameters are also somewhat problematic – while unbiased when fitting the true model with either Bayes or frequentist forms, empirical coverage rates for the between-subjects correlation in intercept parameters, *manifesttraitvar_Y2_Y1*, are well below 95% for all approaches, suggesting caution should be used for any inferences regarding this parameter. The within-subjects correlation parameter, *diffusion_eta2_eta1*, also has problematic coverage rates, but only in the frequentist approach. The substantially over specified, fully random effects Bayesian model, is performing imperfectly, as would be expected, but still quite well. Coverage rates are a little more variable, some mild biases can be seen, and point estimates are a little less precise.

Table 3.2 shows that with only 50 subjects and 10 time points, inferential properties are still reasonable, though somewhat more variable. Some biases are now apparent, with the Bayesian approach showing more than frequentist, as could be expected given that Bayesian approaches typically trade some bias for reduced variance. Compensating for the bias, point estimates from the Bayesian true model are in general much closer to the true value, showing roughly half the RMSE of the frequentist form. While the fully random, over specified Bayesian model is of course somewhat worse in this regard, it still shows much better RMSE than the frequentist approach. Empirical coverage rates from the Bayesian true model are also best, the over specified Bayesian model performs similarly to the frequentist form with Wald type intervals, while surprisingly, the frequentist profile intervals are too often failing to contain the true value.

Table 3.1.: Simulation results comparing Bayesian (B) and frequentist (F) implementations, for 200 subjects and 30 time points. The true mixed-effects dynamic model was fit using both Bayes and frequentist forms, while an over-specified, full random-effects form was also fit with the Bayesian approach. Frequentist intervals were calculated via both profile likelihood, and Wald approaches.

		Coverage				**Mean**				**RMSE**		
Parameter	**Symbol**	**B true**	**B full**	**F profile**	**F Wald**	**True**	**B true**	**B full**	**F**	**B true**	**B full**	**F**
T0mean_eta1	$\boldsymbol{\eta}_{1[1]}$	0.94	0.95	0.95	0.95	1.00	1.00	1.00	1.00	0.09	0.09	0.09
T0mean_eta2	$\boldsymbol{\eta}_{1[2]}$	0.95	0.96	0.94	0.95	1.00	1.00	1.01	1.00	0.10	0.11	0.10
drift_eta1_eta1	$\mathrm{A}_{[1,1]}$	0.95	0.94	0.97	0.96	-0.40	-0.40	-0.42	-0.40	0.04	0.05	0.04
drift_eta2_eta1	$\mathrm{A}_{[1,2]}$	0.95	0.94	0.96	0.96	0.00	0.00	0.01	0.00	0.03	0.04	0.03
drift_eta1_eta2	$\mathrm{A}_{[2,1]}$	0.95	0.93	0.94	0.95	0.10	0.10	0.11	0.10	0.03	0.03	0.03
drift_eta2_eta2	$\mathrm{A}_{[2,2]}$	0.95	0.96	0.95	0.96	-0.20	-0.20	-0.21	-0.20	0.03	0.03	0.03
manifestvar_Y1_Y1	$\Theta_{[1,1]}$	0.95	0.98	0.95	0.96	1.00	1.00	0.99	1.00	0.02	0.03	0.03
manifestvar_Y2_Y2	$\Theta_{[2,2]}$	0.95	0.94	0.95	0.96	1.00	1.00	0.99	1.00	0.02	0.03	0.02
diffusion_eta1_eta1	$\mathrm{Q}_{[1,1]}$	0.97	0.96	0.96	0.96	1.00	1.00	1.01	1.00	0.05	0.06	0.05
diffusion_eta2_eta1	$\mathrm{Q}_{[2,1]}$	0.95	0.95	0.58	0.56	0.45	0.44	0.44	0.45	0.04	0.05	0.04
diffusion_eta2_eta2	$\mathrm{Q}_{[2,2]}$	0.96	0.94	1.00	1.00	1.12	1.12	1.13	1.12	0.04	0.05	0.04
T0var_eta1_eta1	$\mathrm{Q}^*_{1[1,1]}$	0.91	0.61	0.75	0.95	0.50	0.39	0.18	0.47	0.20	0.37	0.16
T0var_eta2_eta1	$\mathrm{Q}^*_{1[2,1]}$	0.99	1.00	0.29	0.60	0.22	0.04	0.04	0.13	0.30	0.20	0.59
T0var_eta2_eta2	$\mathrm{Q}^*_{1[2,2]}$	0.93	0.70	0.64	0.58	0.51	0.38	0.18	0.48	0.23	0.38	0.19
manifestmeans_Y1	$\boldsymbol{\tau}_{[1]}$	0.95	0.97	0.94	0.95	0.50	0.49	0.49	0.50	0.09	0.08	0.09
manifestmeans_Y2	$\boldsymbol{\tau}_{[2]}$	0.94	0.96	0.94	0.95	0.00	-0.00	-0.01	0.00	0.13	0.12	0.12
manifesttraitvar_Y1_Y1		0.94	0.95	0.94	0.94	1.00	1.01	0.98	1.00	0.07	0.07	0.07
manifesttraitvar_Y2_Y1		0.79	0.83	0.42	0.42	0.71	0.71	0.68	0.71	0.05	0.05	0.05
manifesttraitvar_Y2_Y2		0.95	0.94	1.00	0.99	1.41	1.43	1.39	1.41	0.09	0.09	0.09

Table 3.2.: Simulation results comparing Bayesian (B) and frequentist (F) implementations, for 50 subjects and 10 time points. Other details as per Table 3.1

		Coverage				Mean				RMSE		
Parameter	**Symbol**	**B true**	**B full**	**F profile**	**F Wald**	**True**	**B true**	**B full**	**F**	**B true**	**B full**	**F**
T0mean_eta1	$\boldsymbol{\eta}_{1[1]}$	0.94	0.96	0.87	0.94	1.00	1.13	1.04	1.09	0.31	0.29	0.38
T0mean_eta2	$\boldsymbol{\eta}_{1[2]}$	0.95	0.96	0.89	0.96	1.00	1.17	1.04	1.08	0.44	0.37	0.51
drift_eta1_eta1	$\mathrm{A}_{[1,1]}$	0.95	0.88	0.88	0.88	-0.40	-0.55	-0.75	-0.51	0.24	0.41	0.46
drift_eta2_eta1	$\mathrm{A}_{[1,2]}$	0.96	0.95	0.90	0.95	0.00	0.12	0.15	0.06	0.18	0.21	0.44
drift_eta1_eta2	$\mathrm{A}_{[2,1]}$	0.97	0.89	0.90	0.90	0.10	0.17	0.27	0.15	0.13	0.21	0.30
drift_eta2_eta2	$\mathrm{A}_{[2,2]}$	0.95	0.92	0.86	0.94	-0.20	-0.33	-0.41	-0.28	0.17	0.26	0.37
manifestvar_Y1_Y1	$\Theta_{[1,1]}$	0.94	0.95	0.85	0.94	1.00	0.97	0.94	0.94	0.11	0.17	0.19
manifestvar_Y2_Y2	$\Theta_{[2,2]}$	0.95	0.95	0.87	0.97	1.00	0.95	0.92	0.94	0.11	0.15	0.18
diffusion_eta1_eta1	$\mathrm{Q}_{[1,1]}$	0.96	0.95	0.90	0.89	1.00	1.07	1.02	1.09	0.24	0.36	0.35
diffusion_eta2_eta1	$\mathrm{Q}_{[2,1]}$	0.96	0.99	0.42	0.41	0.45	0.38	0.43	0.41	0.17	0.17	0.22
diffusion_eta2_eta2	$\mathrm{Q}_{[2,2]}$	0.95	0.96	0.94	0.95	1.12	1.22	1.20	1.21	0.20	0.26	0.30
T0var_eta1_eta1	$\mathrm{Q}^*_{1[1,1]}$	0.98	0.98	0.69	0.86	0.50	0.45	0.22	0.55	0.26	0.36	0.33
T0var_eta2_eta1	$\mathrm{Q}^*_{1[2,1]}$	1.00	1.00	0.12	0.22	0.22	0.06	0.06	0.11	0.23	0.15	0.86
T0var_eta2_eta2	$\mathrm{Q}^*_{1[2,2]}$	0.99	1.00	0.18	0.26	0.51	0.49	0.25	0.58	0.29	0.35	0.38
manifestmeans_Y1	$\boldsymbol{\tau}_{[1]}$	0.96	0.95	0.90	0.96	0.50	0.37	0.48	0.42	0.29	0.26	0.37
manifestmeans_Y2	$\boldsymbol{\tau}_{[2]}$	0.96	0.96	0.83	0.95	0.00	-0.17	-0.01	-0.07	0.45	0.36	0.52
manifesttraitvar_Y1_Y1		0.96	0.92	0.88	0.97	1.00	1.00	0.93	0.93	0.16	0.19	0.20
manifesttraitvar_Y2_Y1		0.91	0.58	0.13	0.22	0.71	0.66	0.47	0.71	0.13	0.25	0.20
manifesttraitvar_Y2_Y2		0.97	0.96	0.90	0.99	1.41	1.44	1.34	1.35	0.20	0.22	0.24

Table 3.3.: Extended simulation results regarding only the cross-effect parameter. Columns refer to conditions with and without a true cross effect, and with and without varying time intervals, collapsed over N and T conditions. The true mixed-effects model was fit using Bayesian (Bayes true) and Frequentist (including profile, and Wald-type confidence intervals), and a full (over-specified) random-effects model was also fit with the Bayesian form (Bayes full).

	Cross effect = 0		**Cross effect = 0.20**	
Measure	**Equal intervals**	**Varying intervals**	**Equal intervals**	**Varying intervals**
Bayes true coverage	0.96	0.95	0.96	0.95
Bayes full coverage	0.92	0.92	0.92	0.89
Freq. profile coverage	0.96	0.93	0.92	0.92
Freq Wald coverage	0.96	0.95	0.94	0.93
Bayes true mean	0.02	0.02	0.24	0.24
Bayes full mean	0.05	0.05	0.30	0.31
Freq. mean	0.00	0.00	0.24	0.22
Bayes RMSE	0.05	0.05	0.09	0.08
Bayes full RMSE	0.08	0.07	0.13	0.14
Freq. RMSE	0.05	0.05	0.22	0.12
Bayes true CI width	0.22	0.21	0.38	0.36
Bayes full CI width	0.29	0.28	0.45	0.44
Freq. profile CI width	0.23	0.20	0.38	0.34
Freq. Wald CI width	0.21	0.18	0.43	0.38
Bayes true power			0.86	0.86
Bayes full power			0.90	0.92
Freq. profile power			0.82	0.79
Freq. Wald power			0.72	0.74
Bayes true type S	0.04	0.05	0.00	0.00
Bayes full type S	0.08	0.08	0.00	0.00
Freq. profile type S	0.04	0.07	0.00	0.00
Freq. Wald type S	0.04	0.05	0.00	0.00

Table 3.3 shows an extended set of simulation measures, including power to confirm that the true value is not zero and in the correct direction, with a 5% alpha level, and the proportion of type S (Gelman & Carlin, 2014), or sign, errors – the proportion of results where a two-tailed hypothesis test with 5% alpha level would have resulted in an inference that either some effect exists when it does not, or that the effect is in the opposite direction to the true effect. These measures are shown with respect to the cross effect parameter *drift_eta1_eta2* only, for conditions with and without a true cross effect, and with and without varying time intervals. Number of subjects and time points conditions have been collapsed over. The power results show that the Bayesian forms have somewhat better power to confirm a cross effect when one exists, though for the over specified random effects Bayesian model, this will be in part a function of the slightly higher than desirable rate of type S errors in the case of no cross effect – when no cross effect exists, we should only wrongly conclude an effect exists 5% of times.

Table 3.4 shows the same extended set of simulation measures, but distinguishes by number of subjects N, and number of observation time points T, collapsing over varying and non-varying time intervals. These results are primarily to show the change in power and accuracy (RMSE) with varying levels of data collection. In terms of power, all conditions except that with the least data, the $N = 50$ and $T = 30$ condition, showed very good power to reject an incorrect hypothesis of 'no true effect' given that a true cross effect of 0.20 exists. Going up in the number of data points, regardless of whether via more N or more T, seems to improve the RMSE similarly, for these conditions.

Table 3.4.: Extended simulation results regarding only the cross-effect parameter. Columns refer to conditions with and without a true cross effect, with N = 50 or 200 subjects, and T = 10 or 30 time points, collapsed over varying intervals conditions. The true mixed-effects model was fit using Bayesian (Bayes true) and Frequentist (including profile, and Wald, type confidence intervals), and a full (over-specified) random-effects model was also fit with the Bayesian form (Bayes full).

	CR = 0, N = 50		**CR = 0, N = 200**		**CR = 0.2, N = 50**		**CR = 0.2, N = 200**	
Measure	**T = 10**	**T = 30**	**T = 10**	**T = 30**	**T = 10**	**T = 30**	**T = 10**	**T = 30**
Bayes true coverage	0.97	0.94	0.95	0.96	0.97	0.94	0.96	0.94
Bayes full coverage	0.92	0.94	0.92	0.92	0.86	0.94	0.87	0.95
Freq. profile coverage	0.93	0.96	0.96	0.95	0.86	0.93	0.96	0.93
Freq Wald coverage	0.94	0.97	0.96	0.96	0.86	0.95	0.97	0.94
Bayes true mean	0.05	0.01	0.02	0.00	0.29	0.23	0.24	0.20
Bayes full mean	0.12	0.03	0.12	0.01	0.42	0.25	0.31	0.22
Freq. mean	0.01	0.00	0.00	-0.00	0.30	0.21	0.22	0.20
Bayes RMSE	0.10	0.04	0.05	0.02	0.16	0.07	0.08	0.03
Bayes full RMSE	0.16	0.05	0.16	0.02	0.27	0.08	0.14	0.04
Freq. RMSE	0.11	0.04	0.05	0.02	0.48	0.08	0.08	0.03
Bayes true CI width	0.46	0.15	0.18	0.07	0.73	0.28	0.34	0.13
Bayes full CI width	0.64	0.18	0.64	0.07	0.88	0.32	0.44	0.14
Freq. profile CI width	0.45	0.15	0.18	0.07	0.69	0.29	0.35	0.13
Freq. Wald CI width	0.39	0.14	0.18	0.07	0.89	0.28	0.33	0.13
Bayes true power					0.50	0.98	0.95	1.00
Bayes full power					0.68	0.98	0.98	1.00
Freq. profile power					0.34	0.96	0.91	1.00
Freq. Wald power					0.15	0.92	0.83	1.00
Bayes true type S	0.03	0.06	0.05	0.04	0.00	0.00	0.00	0.00
Bayes full type S	0.08	0.06	0.08	0.08	0.00	0.00	0.00	0.00
Freq. profile type S	0.07	0.04	0.04	0.05	0.00	0.00	0.00	0.00
Freq. Wald type S	0.06	0.03	0.04	0.04	0.00	0.00	0.00	0.00

3.4. Dynamics of overall life satisfaction and health satisfaction

To highlight usage of the software and possibilities of the model, we assessed the dynamics of overall life satisfaction and satisfaction with health, for a selection of subjects from the long running German socioeconomic panel (GSOEP) study, using version 29 of the GSOEP data. Questions regarding the fundamental structure of, and causal relations between, subjective wellbeing constructs are still very much open (Busseri & Sadava, 2011; Schimmack, 2008). Dynamic models have been posed as one way of understanding these constructs better (Headey & Muffels, 2014). Given the long time-span over which such constructs are expected to exhibit substantial change — in the order of months, years, or even decades — gathering sufficient data to reasonably fit single-subject models is an unlikely prospect. Further, although the GSOEP is administered yearly, variability in timing of the questionnaire each year results in some variability of time intervals, which if ignored, may add noise and bias. Thus, a hierarchical continuous time approach, in which we leverage variation in the time intervals between questionnaires as additional information, and inform our estimates of specific subjects dynamics based on many other subjects, seems particularly applicable to such data.

3.4.1. Core questions

While many questions might be asked using this approach, the questions we will address here are the very general ones: What are the temporal dynamics of overall and health satisfaction? How much variation in such dynamics exists? Are there relations between cross-sectional age and dynamics, or between certain aspects of dynamics and other aspects?

3.4.2. Sample details

For this example we randomly sampled 200 subjects from the GSOEP that had been observed at all 29 occasions in our data. Such a sub-sample of course no longer benefits from the population representative nature of the GSOEP. This sample resulted in subject ages (calculated at the midpoint of their participation in the study) from 30 to 77 years (mean = 49.23 and sd=10.69). In our subsample, time intervals between measurements ranged from 0.25 to 1.75 years, with a mean of 1 and standard deviation of 0.11.

3.4.3. Constructs

We are interested in the constructs of satisfaction with health, and overall satisfaction with life. These were measured on an 11 point scale. Translations of the questions from German are as follows: "How satisfied are you today with the following areas of your life?" followed by a range of items including "your health". These scales ranged from 0, totally unhappy, to 10, totally happy. Overall satisfaction was assessed separately, as "How satisfied are you with your life, all things considered?" and ranged from 0, completely dissatisfied, to 10, completely satisfied.

3.4.4. Model

The individual level dynamic model was specified as a first order bivariate model. All parameters of the bivariate latent process and measurement models were left free, except for the process intercept and loading matrices. The process intercepts were set to 0, as the measurement model here accounts for non-zero equilibria, and the loading matrix to an identity matrix, for model identification purposes. All free dynamic and measurement model parameters were also free to vary across subjects, with the multivariate population distribution of parameters estimated. R code to generate this model, plot the population distribution priors, and view the resulting Stan code, is provided in Appendix B. The matrix forms for the subject level model are shown in Figure 3.3, with underbraced notations indicating the relevant matrix as described in the model section of this paper, and when appropriate, also the name of the matrix in the ctsem software model specification.

3.4.5. Means of population distributions

Shown in Table 3.5 are the posterior density intervals, point estimates, and diagnostic statistics of the means of the population distributions[3], attained after sampling with six chains, giving potential scale reduction factors Gelman and Rubin, 1992 below 1.05 and more than 100 effective samples per parameter, with a few minor deviations. Note that the median, 50%, is reported as the point estimate, as this is typically closest to the true value in simulations reported in Table 3.1.

Going down the list of parameters shown in Table 3.5, the T0mean parameters are positive for both overall and health satisfaction. Because the T0means represent initial state estimates for the latent processes, and because we have specified the model such that the latent processes have long run means of 0 (by including non-zero manifest means), the population means for the T0mean parameters show to what extent subjects' initial states tend to be higher or lower than their later states. Combining this structure with the positive values observed, suggests that as time goes by, satisfaction scores decline somewhat. Turning to the auto effect parameters of the drift matrix, *drift_healthSat_healthSat* is higher (closer to zero) than *drift_overallSat_overallSat*, suggesting that changes in health satisfaction typically persist longer than changes in overall satisfaction. Sometimes these negative coefficients are confusing

[3]Note that any variance / covariance related parameters are reported as standard deviations and correlations.

$$
\underbrace{\mathrm{d}\begin{bmatrix}\eta_1\\ \eta_2\end{bmatrix}(t)}_{\mathrm{d}\boldsymbol{\eta}(t)} = \left(\underbrace{\begin{bmatrix}\text{drift\_overallSat\_overallSat} & \text{drift\_overallSat\_healthSat}\\ \text{drift\_healthSat\_overallSat} & \text{drift\_healthSat\_healthSat}\end{bmatrix}}_{\underset{\text{DRIFT}}{\mathrm{A}}} \underbrace{\begin{bmatrix}\eta_1\\ \eta_2\end{bmatrix}(t)}_{\boldsymbol{\eta}(t)} + \underbrace{\begin{bmatrix}0\\ 0\end{bmatrix}}_{\underset{\text{CINT}}{\mathrm{b}}} \right) dt +
$$

$$
\underbrace{\begin{bmatrix}\text{diffusion\_overallSat\_overallSat} & 0\\ \text{diffusion\_healthSat\_overallSat} & \text{diffusion\_healthSat\_healthSat}\end{bmatrix}}_{\underset{\text{DIFFUSION}}{\mathrm{G}}} \underbrace{\mathrm{d}\begin{bmatrix}W_1\\ W_2\end{bmatrix}(t)}_{\mathrm{dW}(t)}
$$

$$
\underbrace{\begin{bmatrix}Y_1\\ Y_2\end{bmatrix}(t)}_{\mathrm{Y}(t)} = \underbrace{\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}}_{\underset{\text{LAMBDA}}{\Lambda}} \underbrace{\begin{bmatrix}\eta_1\\ \eta_2\end{bmatrix}(t)}_{\boldsymbol{\eta}(t)} + \underbrace{\begin{bmatrix}\text{manifestmeans\_overallSat}\\ \text{manifestmeans\_healthSat}\end{bmatrix}}_{\underset{\text{MANIFESTMEANS}}{\boldsymbol{\tau}}} + \underbrace{\begin{bmatrix}\epsilon_1\\ \epsilon_2\end{bmatrix}(t)}_{\boldsymbol{\epsilon}(t)}
$$

$$
\underbrace{\begin{bmatrix}\epsilon_1\\ \epsilon_2\end{bmatrix}(t)}_{\boldsymbol{\epsilon}(t)} \sim \mathrm{N}\left(\begin{bmatrix}0\\ 0\end{bmatrix}, \underbrace{\begin{bmatrix}\text{manifestvar\_overallSat\_overallSat} & 0\\ 0 & \text{manifestvar\_healthSat\_healthSat}\end{bmatrix}}_{\underset{\text{MANIFESTVAR}}{\Theta}} \right)
$$

Figure 3.3.: Matrix specification of the subject level model of overall life satisfaction and satisfaction with health. Underbraced notations denoting the symbol used to represent the matrix in earlier formulas, and where appropriate also the matrix name in the ctsem specification[a].

[a] Strictly speaking, the diffusion matrix is actually the covariance matrix $\mathrm{GG}^\top$, but G is the way it is specified in ctsem.

for those used to discrete-time results, but they provide a relatively intuitive interpretation – the higher above baseline a process is, the stronger the downwards pressure due to the auto-effect, and the further below baseline, the stronger the upwards pressure. The cross effect parameter *drift_healthSat_overallSat* is very close to zero, suggesting that changes in overall satisfaction do not predict later changes in health satisfaction. Conversely however, *drift_overallSat_healthSat* is substantially positive, so changes in health satisfaction are predictive of later changes in overall satisfaction, in the same direction. To understand these temporal dynamics due to the drift matrix more intuitively, the expected auto and cross regressions over time are shown in Figure 3.4, where we see for instance that the expected effect of health satisfaction on overall satisfaction peaks at time intervals of around 2-4 years. Note that this does *not* imply that the effect is changing with time – but because processes are somewhat stable, and a change at one time (in the plot, a change of 1.00 at time zero) persists, this allows for consequences of the change to continue building for some time. The diffusion matrix correlation parameter,

Table 3.5.: Posterior intervals and point estimates for means of estimated population distributions

	symbol	2.5%	50%	97.5%
T0mean_overallSat	$\boldsymbol{\eta}_{1[1]}$	0.59	0.86	1.13
T0mean_healthSat	$\boldsymbol{\eta}_{1[2]}$	1.19	1.50	1.81
drift_overallSat_overallSat	$A_{[1,1]}$	-0.61	-0.39	-0.25
drift_overallSat_healthSat	$A_{[1,2]}$	0.01	0.09	0.21
drift_healthSat_overallSat	$A_{[2,1]}$	-0.06	-0.01	0.06
drift_healthSat_healthSat	$A_{[2,2]}$	-0.30	-0.19	-0.12
diffusion_overallSat_overallSat	$Q_{[1,1]}$	0.52	0.67	0.83
diffusion_healthSat_overallSat	$Q_{[2,1]}$	0.72	0.84	0.92
diffusion_healthSat_healthSat	$Q_{[2,2]}$	0.50	0.60	0.72
manifestvar_overallSat_overallSat	$\Theta_{[1,1]}$	0.79	0.86	0.92
manifestvar_healthSat_healthSat	$\Theta_{[2,2]}$	0.91	0.98	1.04
manifestmeans_overallSat	$\boldsymbol{\tau}_{[1]}$	6.71	6.90	7.07
manifestmeans_healthSat	$\boldsymbol{\tau}_{[2]}$	6.06	6.33	6.54
T0var_overallSat_overallSat	$Q^*_{1[1,1]}$	0.01	0.25	1.39
T0var_healthSat_overallSat	$Q^*_{1[2,1]}$	-0.62	0.33	0.86
T0var_healthSat_healthSat	$Q^*_{1[2,2]}$	0.02	0.29	0.98

diffusion_healthSat_overallSat, suggests that the random disturbances impacting each process are quite positively correlated, which is unsurprising as we might expect that overall and health satisfaction certainly share some common causes. The two diffusion matrix standard deviation parameters are difficult to interpret on their own because they do not reflect total variance in the system, but rather only the rate of incoming variance (unexplained exogenous inputs). Instead we can consider the asymptotic innovation variances – the diagonal of $Q^*_{\Delta t=\infty}$ – or in other words, the expected total latent process variance. These are 0.42 for overall satisfaction, and 0.24 for health satisfaction. This implies that the latent overall satisfaction process has more variability over time. The two manifest indicators show similar standard deviations for measurement error (manifestvar_overallSat_overallSat and manifestvar_healthSat_healthSat) for each process, which implies that measurement limitations and short term situational influences (e.g., a sunny day) contribute similar levels of variance to each indicator. Further, it seems that such influences are a larger source of variance for observed scores than the latent process variability – given that we fixed factor loadings to 1.00, they are directly comparable. This is not however suggestive that the measures are unreliable per se, as the measurement error and total latent process variance only reflect within-person variability – to consider reliability one would need to also consider the between-person variance in baseline levels, accounted for in this model by the manifest means. The manifest means parameters reflect the intercepts of the manifest indicators, and here both are at roughly similar levels, between 6 and 7. The absolute value here is probably not so interesting, it is rather the individual differences, and relations between individual differences and other parameters or covariates, that are of most interest, and these are discussed later. The T0var parameters reflect the initial variance and covariance of the latent processes, and there is generally high uncertainty associated with the estimation of such parameters.

3.4.6. Variance of population distributions

Shown in Table 3.6 are the posterior density intervals of standard deviation parameters of the population distributions, showing to what extent individual subjects parameter values tended to differ from the population mean values, in ways that could not be predicted by our age covariates – the unexplained between-subjects variance in a parameter. So while every subject has their own particular set of parameters, the estimated mean of the parameter distribution over all subjects is shown in Table 3.5, and the standard deviation is shown in Table 3.6. Individual differences in the T0means is unsurprising, as they simply reflect differences in the

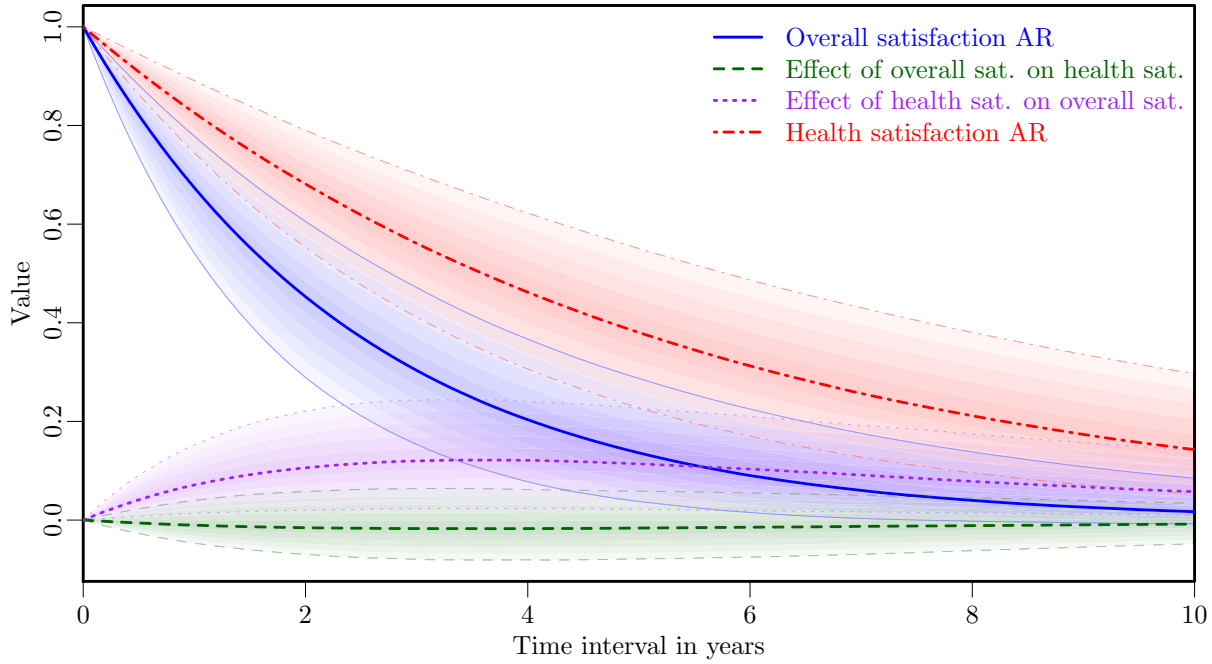


Figure 3.4.: Median and 95% quantiles of auto and cross regressions over time.

initial level of the latent process states. Looking at the temporal dynamics parameters, both auto effects (*drift_overallSat_overallSat* and *drift_healthSat_healthSat*) show some variability, reflecting individual differences in the persistence of changes in overall and health satisfaction processes. Regarding cross effects, there are quite some differences in variability, which would seem consistent with the strength of the effects at the population level. That is, the non-existent or very weak average effect of overall satisfaction on health shows little variability, while the stronger effect of health satisfaction on later overall satisfaction varies more across subjects – the effect may be important, but more so for some people than others. Just as with the mean of the population distribution of diffusion parameters, the between-subjects standard deviations of diffusion parameters (*diffusion_overallSat_overallSat* and *diffusion_healthSat_healthSat*) are not so straightforward to interpret. This is because the effect of the diffusion parameters is dependent on the also varying drift parameters, and instead, one could consider variance in the asymptotic diffusion parameters – this would also be the case for a discrete-time analysis. We can however interpret between-subjects variability in the diffusion correlation parameter (*diffusion_healthSat_overallSat*), which is relatively modest, suggesting that the bulk of subjects show the positive correlations indicated by the mean of the population distribution, as discussed earlier. The between-subjects variability in measurement error (*manifestvar*) parameters is moderate, which may reflect that some people respond to the GSOEP with more randomness, or more influence of transient conditions. Between-subjects variation in the manifest means parameters reflects individual differences in baseline levels of the two satisfaction processes, and we see slightly more variation for health satisfaction here.

3.4.7. Correlations in individual differences

In order to better understand the finding that changes in health satisfaction appear to predict later changes in overall satisfaction, we also investigated correlates of individual differences in that effect, in the other model parameters. These results are shown in Table 3.7.

The strongest correlations between the effect of health satisfaction on later overall satisfaction, and other subject level parameters, are the two correlations with the auto effects of

Table 3.6.: Posterior distributions for standard deviation of population distributions

	2.5%	50%	97.5%
T0mean_overallSat	0.21	1.21	1.57
T0mean_healthSat	0.43	1.13	1.49
drift_overallSat_overallSat	0.17	0.29	0.46
drift_overallSat_healthSat	0.00	0.08	0.22
drift_healthSat_overallSat	0.00	0.02	0.06
drift_healthSat_healthSat	0.09	0.16	0.26
diffusion_overallSat_overallSat	0.29	0.38	0.51
diffusion_healthSat_overallSat	0.13	0.22	0.37
diffusion_healthSat_healthSat	0.20	0.28	0.39
manifestvar_overallSat_overallSat	0.26	0.30	0.36
manifestvar_healthSat_healthSat	0.30	0.36	0.41
manifestmeans_overallSat	0.83	0.95	1.09
manifestmeans_healthSat	1.10	1.27	1.45
T0var_overallSat_overallSat	0.02	0.24	0.74
T0var_healthSat_overallSat	0.01	0.31	1.09
T0var_healthSat_healthSat	0.05	0.35	0.69

the drift matrix, the two with the diffusion standard deviations, and those with the baseline levels of the processes, the manifestmeans parameters. Interpretations of correlations between one temporal dynamics parameter and other parameters of the drift matrix is not straightforward, as changes in one parameter can to some extent be compensated by changes in another, resulting in some level of correlation that is likely inherent to the model structure. However there is no such possibility in regards to the correlations with the baseline level parameters, *manifestmeans_healthSat* and *manifestmeans_overallSat*. These suggest that for subjects with reduced health satisfaction in general, changes in health satisfaction are more predictive of later overall satisfaction. Though weaker, there is some evidence for a similar effect in relation to baseline levels of overall satisfaction as well. Cross-lagged effects have typically been considered important due to the improved possibilities for causal inference, this finding gives some insight as to factors that may influence the causal structure of wellbeing constructs.

3.4.8. Covariate effects

Cross-sectional age and age squared were included as time independent predictors for all parameters. Appendix C contains a table of full results, but because we are looking at a quadratic effect, examining the multi-dimensional credible region is much simpler with the plots of Figure 3.5 (generated using the ctStanTIpredeffects function from ctsem). This figure shows the 50% credible intervals for the quadratic effect of age on the model parameters. 50% was chosen for the sake of interpretability of plots in this example, and we do not mean for this interval to be taken as strong support for any interpretations – although the estimated effects are regularised by the standard normal prior, to reduce the chance of large spurious effects. The first plot focuses on initial and average levels of the processes, with the strongest effect being that the average level of health satisfaction (*manifestmeans_healthSat*) declines with age. This is coupled with a rise in the initial state of health satisfaction (*T0means_healthSat*) relative to baseline, which simply means that older subjects are showing more downwards trend over the range of observations. In this case, the within-person effect indicated by t0means suggests that older adults show steeper declines, while the estimated between-person effect based on the manifestmeans parameter suggests some levelling off. We would not make much of such a discrepancy at this point, for a more complete modelling of trends an additional latent process with zero diffusion could be included. There appears to be little effect of age on overall satisfaction (*manifest_means_overallSat*), except the initially observed levels (*T0mean_overallSat*)

Table 3.7.: Posterior distributions for correlations between effect of health satisfaction on later overall satisfaction (*drift_overallSat_healthSat*), and all other subject level parameters.

	2.5%	50%	97.5%
drift_overallSat_overallSat	-0.72	-0.44	0.00
manifestmeans_overallSat	-0.44	-0.12	0.19
manifestmeans_healthSat	-0.52	-0.16	0.24
T0var_healthSat_healthSat	-0.54	-0.14	0.30
diffusion_healthSat_healthSat	-0.53	-0.15	0.30
drift_healthSat_overallSat	-0.77	-0.18	0.48
manifestvar_overallSat_overallSat	-0.34	-0.03	0.33
T0mean_overallSat	-0.39	-0.01	0.35
T0var_healthSat_overallSat	-0.72	-0.03	0.68
diffusion_healthSat_overallSat	-0.44	0.03	0.47
T0mean_healthSat	-0.37	0.03	0.42
T0var_overallSat_overallSat	-0.42	0.09	0.68
manifestvar_healthSat_healthSat	-0.22	0.07	0.41
diffusion_overallSat_overallSat	-0.22	0.12	0.47
drift_healthSat_healthSat	-0.15	0.35	0.73

may be somewhat raised in the younger and older subjects.

Looking at the second plot containing temporal dynamics parameters from the drift matrix, the largest effect seems to be a decline in the persistence of changes in overall satisfaction (*drift_overallSat_overallSat*) during mid-life, and a rise with older ages. There is also a decrease in the effect of health satisfaction on overall satisfaction in older ages. Turning to the third plot containing the diffusion parameters, younger subjects appear to show somewhat higher within-subject variability in overall satisfaction (*diffusion_overallSat_overallSat*). In the case of health satisfaction, it is instead the middle aged subjects showing more within-subject variability (*diffusion_healthSat_healthSat*). With increasing age, random (in the sense that the model does not predict them) changes to health and overall satisfaction appear to become more correlated (*diffusion_healthSat_overallSat*). Finally, both overall and health satisfaction measurements appear to become a little more error prone with age, as shown in the final plot.

3.4.9. Individual level analyses

For individual level analysis, we may compute predicted (based on all prior observations), updated (based on all prior and current observations), and smoothed (based on all observations) expectations and covariances from the Kalman filter, based on specific subjects models. All of this is readily achieved with the ctStanKalman function from ctsem. This approach allows for: predictions regarding individuals‘ states at any point in time, given any values on the time dependent predictors (external inputs such as interventions or events); residual analysis to check for unmodeled dependencies in the data; or simply as a means of visualization, for comprehension and model sanity checking purposes. An example of such is depicted in Figure 3.6, where we see observed and smoothed scores with uncertainty, for a randomly selected subject from our sample.

3.5. Discussion

We have described a hierarchical continuous time dynamic model, for the analysis of repeated measures data of multiple subjects. In addition, we have introduced a Bayesian extension to the free and open-source software ctsem (Driver et al., 2017) that allows fitting this model. The

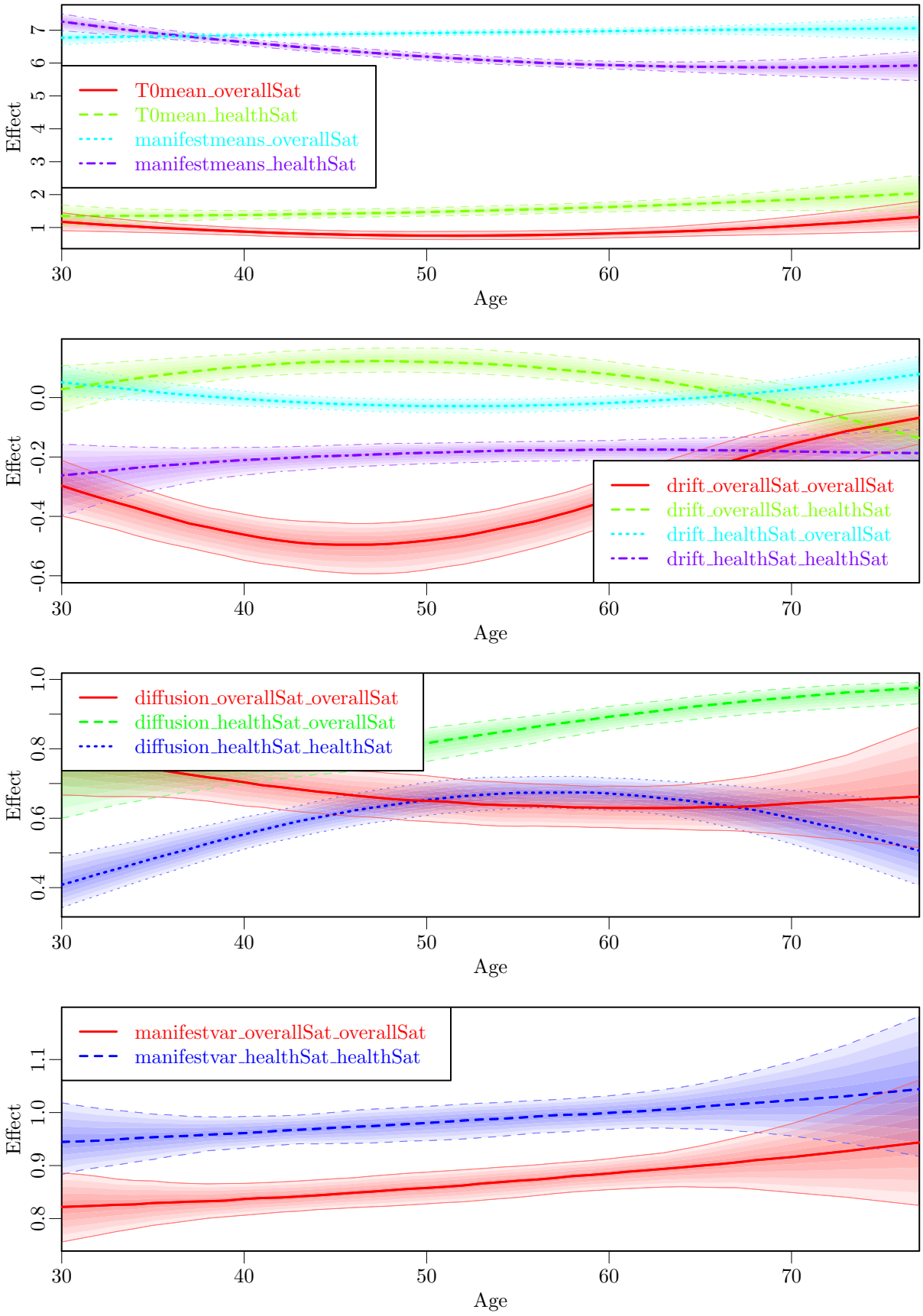


Figure 3.5.: Estimated effect of age on model parameters.

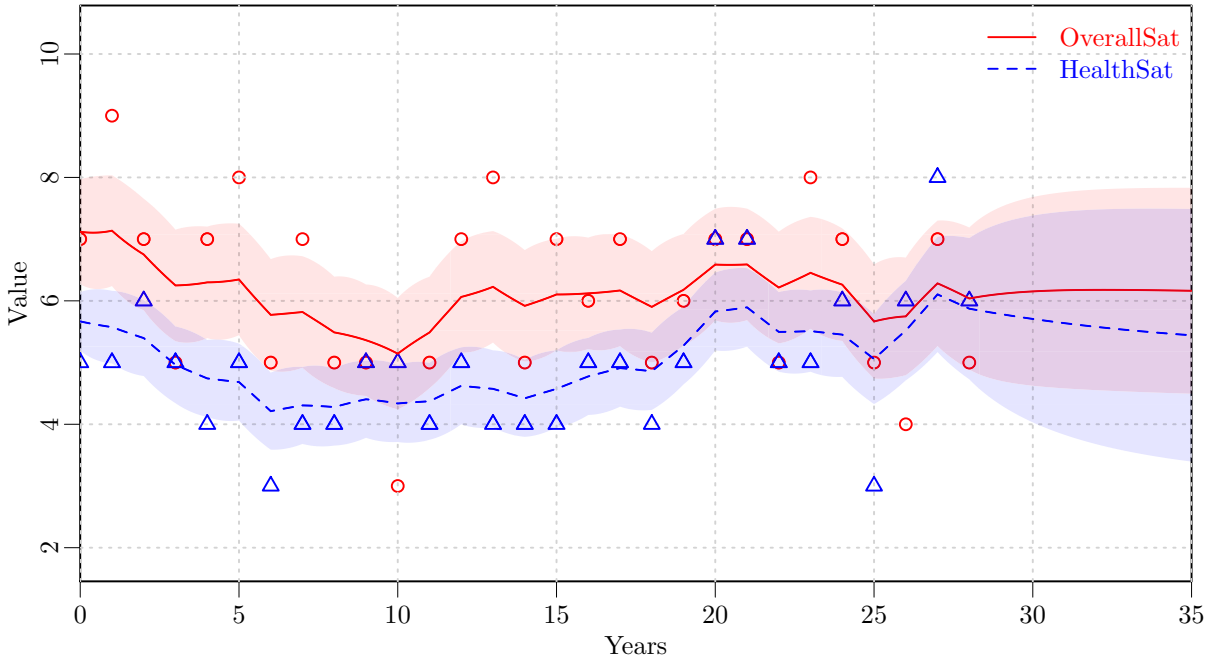


Figure 3.6.: Observed scores for a single subject and Kalman smoothed score estimates from the subjects dynamic model.

subject level model is flexible enough to allow for many popular longitudinal model structures in psychological research to be specified, including forms of latent trajectories (Delsing & Oud, 2008), autoregressive cross-lagged models (and thus the latent change score formulation, see Voelkle & Oud, 2015), higher order oscillating models, or some mixture of approaches. We can use the model to learn about the measurement structure, variance and covariance of the latent processes stochastic aspects, and temporal dynamics of the processes. Because it is a continuous time model, there are no restrictions on the timing of data collection. Time intervals between observations can vary both within and across individuals, and indeed such variability is likely to improve the estimation of the model (Voelkle & Oud, 2013). The hierarchical aspect ensures that while each individual can have their own distinct model parameters, data collected from other subjects is still informative, leading generally to improved individual and population estimates. The inclusion of subject specific covariates (time independent predictors) may be used to improve estimation and inform about relationships to parameters, but is also not necessary as any sort of control, as heterogeneity at the individual level is accounted for by allowing random subject specific parameters.

Results from a limited simulation study suggest that the Bayesian form offers reliable inferential properties when the correct model is specified, with only marginal reductions when an overly complex, full random effects population model is specified. In the more limited data conditions, point estimates were generally much closer to the true value using the Bayesian specification, at cost of some additional bias – though coverage rates with the Bayesian model remained good, in many cases better than the frequentist form. Such a change in the bias and variance trade-off is not unexpected with a switch from frequentist to Bayesian forms, though the magnitude of improvement in the RMSE, in some cases roughly halved, was surprising to us, given that the priors were left in the general state we proposed and. Of course, while the generating model was largely based on the empirical results from the GSOEP data, if the generating model had parameter values that differed substantially from what we have posed as a rough norm (see Figure 3.2) the story may be quite different. Thus, care should of course be taken that variables are scaled and centred, and the time scale is such that neither extremely

high or extremely low auto effects are expected. For instance, in the empirical example, if we had assessed the time intervals in the GSOEP data in terms of seconds instead of years, auto effects would have been extremely close to the upper boundary, zero – satisfaction simply does not change that quickly, so far as we are aware! While we believe the model and software are ready for reliable inference for most uses, coverage rates for confidence intervals of between subject correlations appear to be sub-par, though no substantial bias is evident. This is also a problem in the frequentist approaches examined, and is at present unclear to what extent it is a problem inherent to the form of model, or a limitation in our specification. More comprehensive power and simulation studies will help to improve the understanding of this and similar models under a wider range of conditions, the intent of our results is to give some confidence in the procedure, and highlight any cases where caution may be warranted.

Chapter 4.

Hierarchical Bayesian Continuous Time Dynamic Modelling With ctsem

Version 1 of ctsem provided structural equation model based functionality by linking to the OpenMx software, allowing mixed effects models (random intercepts but fixed regression and variance parameters) for multiple subjects. For version 2 of the R package ctsem, we include a Bayesian specification and fitting routine that uses the Stan probabilistic programming language, via the rstan package in R. This allows for all parameters of the dynamic model to individually vary, based on an estimated population mean and variance, and any time independent covariate effects. Frequentist approaches with ctsem are documented in Chapter 2, and in R vignette form at https://cran.r-project.org/package=ctsem/vignettes/ctsem.pdf. The Bayesian approach is discussed in full in Chapter 3, here we provide specifics on the usage of the ctsem software for hierarchical Bayesian approaches. For the most up to date version of this document, see https://cran.r-project.org/package=ctsem/vignettes/hierarchical.pdf

4.1. Overview of hierarchical model

The model is described in full in Chapter 3, but a brief summary of the approach is as follows: Parameters of the measurement and dynamic models for each subject are first drawn from a simultaneously estimated higher level distribution over an unconstrained space, then a set of parameter specific transformations are applied so that a) each parameter conforms to necessary bounds and b) is subject to the desired prior. Then, a range of matrix transformations are applied to generate the continuous time parameter matrices, as well as the necessary discrete time instantiations, given the time intervals observed (More variability in measurement time intervals thus means more computations). The higher level distribution has a multivariate normal prior.

4.2. Install software and prepare data

Install ctsem software:

```
install.packages("ctsem")
```

Prepare data in long format, each row containing one time point of data for one subject. We need a subject id column containing numbers from 1 to total subjects, rising incrementally with each subject going down the data structure. This is to ensure coherence with the internal structure of the Stan model. The column is named by default "id", though this can be changed in the model specification. We also need a time column "time", containing numeric values for time, columns for manifest variables (the names of which must be given in the next step using ctModel), columns for time dependent predictors (these vary over time but have no model estimated and are assumed to impact latent processes instantly), and columns for time

independent predictors (which predict the subject level parameters, that are themselves time invariant – thus the values for a particular time independent predictor must be the same across all observations of a particular subject).

```
     id  time        Y1         Y2         TD1       TI1        TI2       TI3
[1,]  1 -0.04 -5.620461 -11.517371  0.47076466 0.5287038 -0.9917957 0.8370261
[2,]  1  1.08 -4.206223 -10.394572  0.82351520 0.5287038 -0.9917957 0.8370261
[3,]  1  1.87 -4.648091 -10.356792  0.12355094 0.5287038 -0.9917957 0.8370261
[4,]  1  3.19 -4.345848  -9.917012 -1.11548038 0.5287038 -0.9917957 0.8370261
[5,]  1  3.95 -5.244832 -10.529313 -0.06299107 0.5287038 -0.9917957 0.8370261
[6,]  2 35.99 -1.014329 -17.119419  0.37658066 1.1378486  0.3234105 0.1470207
[7,]  2 37.04 -2.102099 -17.645759 -0.15865655 1.1378486  0.3234105 0.1470207
[8,]  2 37.87 -2.534922 -17.691342  0.90665532 1.1378486  0.3234105 0.1470207
```

At present, missingness is fine on manifest indicators, but not allowed elsewhere.

4.3. Model specification

Specify model using `ctModel(type="stanct",...)`. "stanct" specifies a continuous time model in Stan format, "standt" specifies discrete time, while "omx" is the classic ctsem behaviour and prepares an OpenMx model. Other arguments to ctModel proceed as normal, although some matrices used for type 'omx' are not relevant for the Stan formats, either because the between subject matrices have been removed, or because time dependent and independent predictors are now treated as fixed regressors and only require effect (or design) matrices. These differences are documented in the help for ctModel.

```
model<-ctModel(type="stanct",
  n.latent=2, latentNames=c("eta1","eta2"),
  n.manifest=2, manifestNames=c("Y1","Y2"),
  n.TDpred=1, TDpredNames="TD1",
  n.TIpred=3, TIpredNames=c("TI1","TI2","TI3"),
  LAMBDA=diag(2))
```

This generates a first order bivariate latent process model, with each process measured by a single, potentially noisy, manifest variable. A single time dependent predictor is included in the model, and three time independent predictors. Additional complexity or restrictions may be added, the table below shows the basic arguments one may consider and their link to the dynamic model parameters. For more details see the ctsem help files or papers. Note that for the Stan implementation, ctModel requires variance covariance matrices (DIFFUSION, T0VAR, MANIFESTVAR) to be specified with standard deviations on the diagonal, correlations (partial, if > 2 latent processes) the lower off diagonal, and zeroes on the upper off diagonal.

These matrices may all be specified using a combination of character strings to name free parameters, or numeric values to represent fixed parameters.

One may modify the output model to either restrict between subject differences (set some parameters to not vary over individuals), alter the transformation used to determine the prior / bounds, or restrict which effects of time independent predictors to estimate. Plotting the original prior, making a change, and plotting the resulting prior, are shown here – in this case we believe the stochastic latent process innovation for our first latent process, captured by row 1 and column 1 of the DIFFUSION matrix, to be small, so scale our prior accordingly to both speed and improve sampling. Rather than simply scaling by 0.2 as shown here, one could also construct a new form of prior, so long as the resulting distribution was within the bounds required for the specific parameter. Note that the resulting distribution is a result of applying the specified transformation to a standard normal distribution, with mean of 0 and standard

Table 4.1.: Command line arguments for the ctModel function, when building a hierarchical Bayesian continuous time dynamic model.

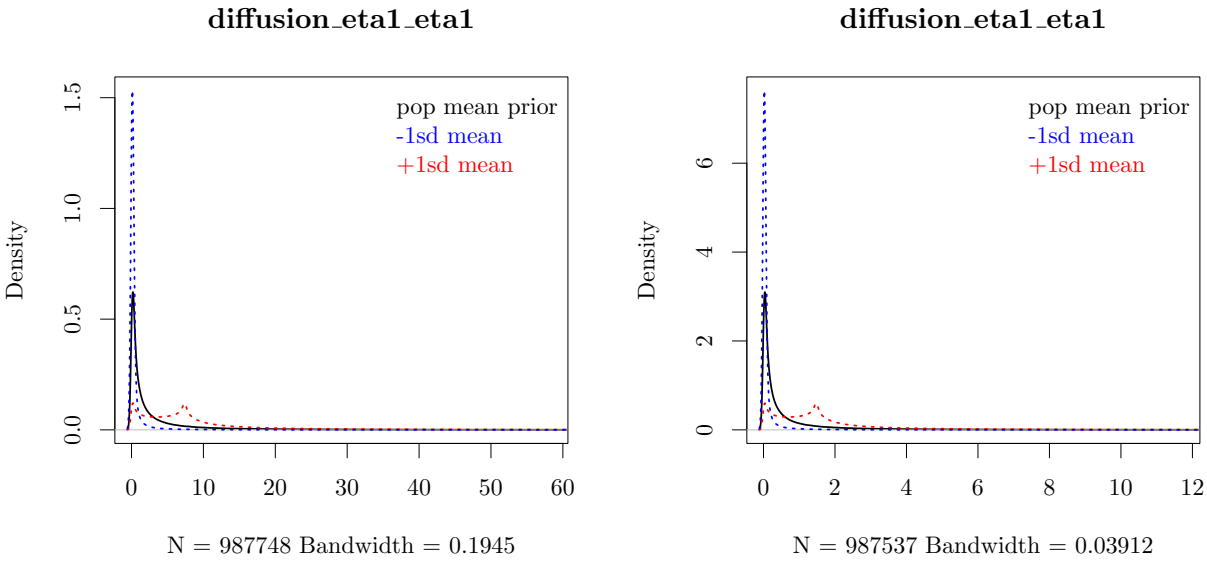
Argument	**Sign**	**Default**	**Meaning**
n.manifest	c		Number of manifest indicators per individual at each measurement occasion.
n.latent	v		Number of latent processes.
LAMBDA	Λ		n.manifest × n.latent loading matrix relating latent to manifest variables.
manifestNames		Y1, Y2, etc	n.manifest length character vector of manifest names.
latentNames		eta1, eta2, etc	n.latent length character vector of latent names.
T0VAR	Q_1^*	free	lower tri n.latent × n.latent matrix of latent process initial covariance, specified with standard deviations on diagonal and (partial) correlations on lower triangle.
T0MEANS	η_1	free	n.latent × 1 matrix of latent process means at first time point, T0.
MANIFESTMEANS	τ	free	n.manifest × 1 matrix of manifest means.
MANIFESTVAR	Θ	free diag	lower triangular matrix of var / cov between manifests, specified with standard deviations on diagonal and (partial) correlations on lower triangle.
DRIFT	A	free	n.latent × n.latent matrix of continuous auto and cross effects.
CINT	b	0	n.latent × 1 matrix of continuous intercepts.
DIFFUSION	Q	free	lower triangular n.latent × n.latent matrix containing standard deviations of latent process on diagonal, and (partial) correlations on lower off-diagonals.
n.TDpred	l	0	Number of time dependent predictors in the dataset.
TDpredNames		TD1, TD2, etc	n.TDpred length character vector of time dependent predictor names.
TDPREDEFFECT	M	free	n.latent × n.TDpred matrix of effects from time dependent predictors to latent processes.
n.TIpred	p	0	Number of time independent predictors.
TIpredNames		TI1, TI2, etc	n.TIpred length character vector of time independent predictor names.

Table 4.2.: The pars subobject of the created model object shows the parameter specification that will go into Stan, including both fixed and free parameters, whether the parameters vary across individuals, how the parameter is transformed from a standard normal distribution (thus setting both priors and bounds), and whether that parameter is regressed on the time independent predictors.

```
head(model$pars,8)
   matrix row col           param value                      transform indvarying
1 TOMEANS   1   1     TOmean_eta1    NA                    (param) * 10       TRUE
2 TOMEANS   2   1     TOmean_eta2    NA                    (param) * 10       TRUE
3  LAMBDA   1   1            <NA>     1                           <NA>      FALSE
4  LAMBDA   1   2            <NA>     0                           <NA>      FALSE
5  LAMBDA   2   1            <NA>     0                           <NA>      FALSE
6  LAMBDA   2   2            <NA>     1                           <NA>      FALSE
7   DRIFT   1   1 drift_eta1_eta1    NA -log(exp(-param*1.5)+1)-.00001       TRUE
8   DRIFT   1   2 drift_eta1_eta2    NA                     (param)*.5       TRUE
  sdscale TI1_effect TI2_effect TI3_effect
1       1       TRUE       TRUE       TRUE
2       1       TRUE       TRUE       TRUE
3       1      FALSE      FALSE      FALSE
4       1      FALSE      FALSE      FALSE
5       1      FALSE      FALSE      FALSE
6       1      FALSE      FALSE      FALSE
7       1       TRUE       TRUE       TRUE
8       1       TRUE       TRUE       TRUE
```

deviation of 1. To change the underlying standard normal, one would need to edit the resulting Stan code directly.

```
par(mfrow=c(1,2))
plot(model,rows=11)
print(model$pars$transform[11])
[1] "exp(param*2) +.00001"
model$pars$transform[11]<- "(exp(param*2) +.0001)*.2"
plot(model,rows=11)
```



The plots show the prior distribution for the population mean of DIFFUSION[1,1] in black, as well as two possible priors for the subject level parameters. The blue prior results from assuming the population mean is one standard deviation lower than the mean of the prior, and

marginalising over the prior for the standard deviation of the population distribution. This latter prior can be scaled using the sdscale column of the parameters subobject, but is by default a truncated normal distribution with mean 0 and SD 1.

Restrict between subject effects as desired. Unnecessary between subject effects will slow sampling and hinder appropriate regularization, but be aware of the many parameter dependencies in these models – restricting one parameter may lead to genuine variation in the restricted parameter expressing itself elsewhere. The prior scale for between subject variance may need to be restricted when limited data (in either the number of time points or number of subjects) is available, to ensure adequate regularisation. Here we restrict MANIFESTVAR effects between subjects, and set all prior scales for the standard deviation of the population distribution to 0.2, from the default of 1.0. A rough interpretation of this change in sdscale is simply that we expect lower values for the population standard deviation, but to better interpret the effect of this latter change, see the section on standard deviation transformations.

```
model$pars[c(15,18),]$indvarying<-FALSE
model$pars$sdscale[1:28] <- .5
```

Also restrict which parameters to include time independent predictor effects for in a similar way, for similar reasons. In this case, the only adverse effects of restriction are that the relationship between the predictor and variables will not be estimated, but the subject level parameters themselves should not be very different, as they are still freely estimated. Note that such effects are only estimated for individually varying parameters anyway – so after the above change there is no need to set the tipredeffect to FALSE for the T0VAR variables, it is assumed. Instead, we restrict the tipredeffects on all parameters, and free them only for the manifest intercept parameters.

```
model$pars[,c("TI1_effect","TI2_effect","TI3_effect")]<-FALSE
model$pars[c(19,20),c("TI1_effect","TI2_effect","TI3_effect")]<-TRUE
```

4.4. Model fitting

Once model specification is complete, the model is fit to the data using the ctStanFit function as follows – depending on the data, model, and number of iterations requested, this can take anywhere from a few minutes to days. Current experience suggests 300 iterations is often enough to get an idea of what is going on, but more may be necessary for robust inference. This will of course vary massively depending on model and data. For the sake of speed for this example we only sample for 200 iterations with a lowered max_treedepth - this latter parameter controls the maximum number of steps the Hamiltonian sampler is allowed to take per iteration, with each increase of 1 doubling the maximum. With these settings the fit should take only a few minutes (but will not be adequate for inference!). Those that wish to try out the functions without waiting, can simply use the already existing ctstantestfit object instead of creating the fit object (and adjust the code in following sections as needed).

```
fit<-ctStanFit(datalong = ctstantestdat, ctstanmodel = model, iter=200,
  chains=2, plot=FALSE, control=list(max_treedepth = 6))
```

The plot argument allows for plotting of sampling chains in real time, which is useful for slow models to ensure that sampling is proceeding in a functional manner. Models with many parameters (e.g., many subjects and all parameters varying over subject) may be too taxing for the function to handle smoothly - we have had success with up to around 4000 parameters.

4.5. Summary

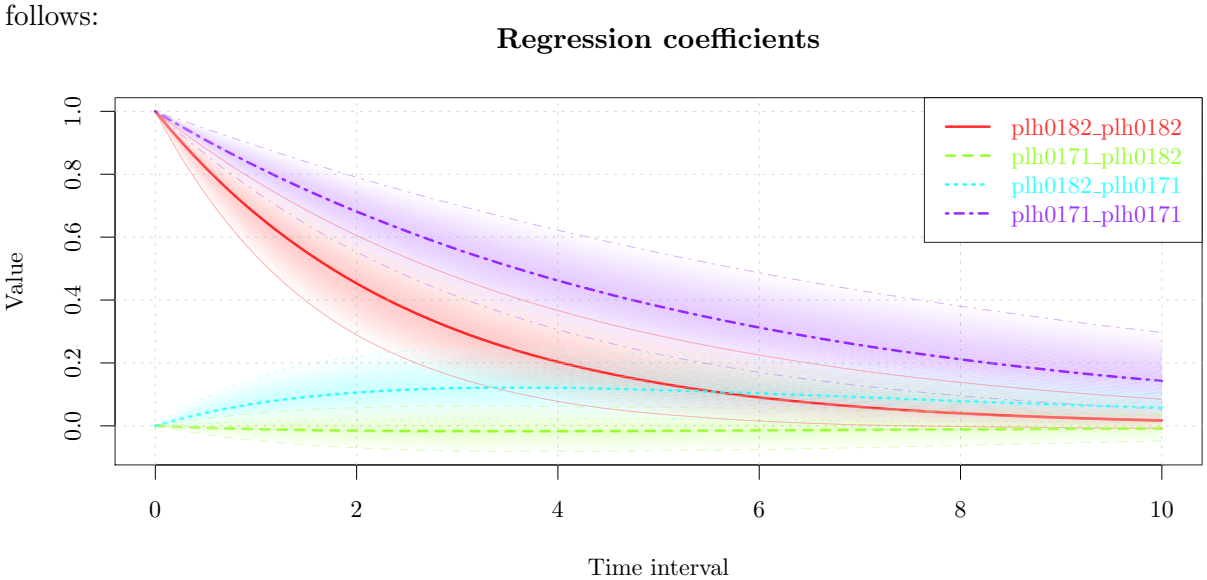
After fitting, the summary function may be used on the fit object, which returns details regarding the population mean parameters, population standard deviation parameters, population correlations, and the effect parameters of time independent predictors. Additionally, summary outputs a range of matrices regarding correlations between subject level parameters. hypercorr_means reports the posterior mean of the correlation between raw (not yet transformed from the standard normal scale) parameters. hypercorr_sd reports the standard deviation of these parameters. hypercovcor_transformedmean reports the correlation between transformed parameters on the lower triangle, the variance of these parameters on the diagonal, and the covariance on the upper triangle. To view the posterior median of the continuous time parameter matrices, the ctStanContinuousPars function can be used.

```
summary(fit)
```

In the summary output, the population parameters are returned in the same form that they are input to ctModel - that is, standard deviations and partial correlations, which can be more difficult to interpret when more than two processes are modeled. To return the full set of continuous time parameter matrices for a specified subject or collection of subjects, the ctStanContinuousPars function can be used, with the calcfunc argument set appropriately depending on whether one wants the median,mean, sd, or a specific quantile (which would require changing the probs argument also).

4.6. Plotting

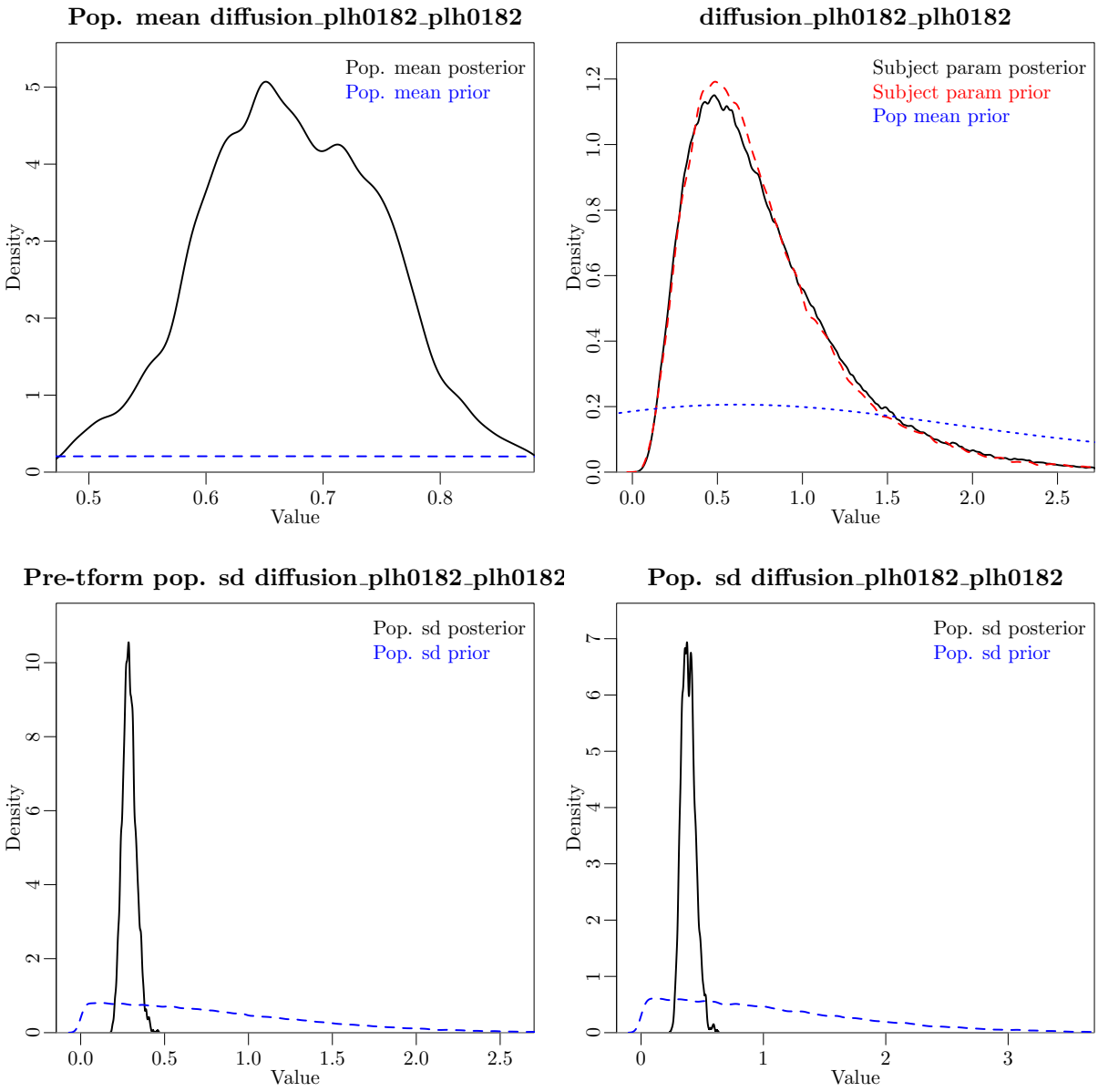
The plot function outputs a sequence of plots, all generated by specific functions. The name of the specific function appears in a message in the R console, checking the help for each specific function and running them separately will allow more customization. Some of the plots, such as the trace, density, and interval, are generated by the relevant rstan function and hopefully self explanatory. The plots specific to the heirarchical continuous time dynamic model are as follows:



The above plot shows the dynamic regression coefficients (between latent states at different time points) that are implied by the model for particular time intervals, as well as the uncertainty of these coefficients.

The relation between posteriors and priors for variables of interest can also be plotted as follows:

```
ctStanPlotPost(ctstanfitobj = fit, rows=11, mfrow = c(2, 2))
```



Shown are approximate density plots based on the post-warmup samples drawn. For each parameter four plots are shown – the first displays the posterior distribution of subject level parameters, the subject level prior (generated from repeated sampling of the hyper parameters), and the prior for the population mean.

4.7. Stationarity

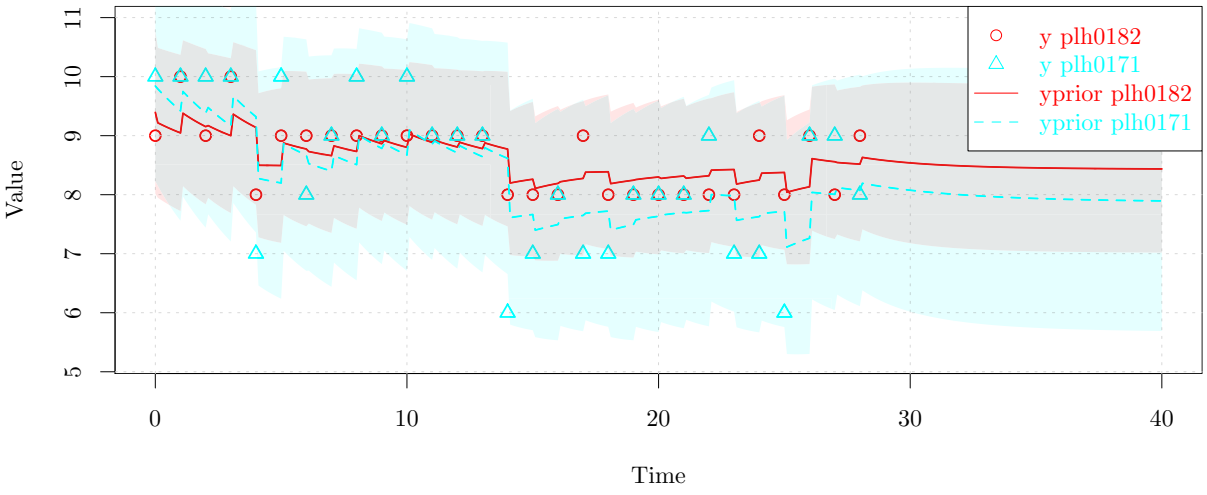
When it is reasonable to assume that the prior for long term expectation and variance of the latent states is the same as (or very similar to) the prior for initial expectations and variances, setting the argument stationary=TRUE to ctStanFit can reduce uncertainty and aid in fitting. This argument ignores any T0VAR and T0MEANS matrices in the input, instead replacing them with asymptotic expectations based on the DRIFT, DIFFUSION, and CINT matrices. Alternatively, a prior can be placed on the stationarity of the dynamic models, calculated as the difference between the T0MEANS and the long run asymptotes of the expected value of the process, as well as the difference between the diagonals of the T0VAR covariance matrix and the

long run asymptotes of the covariance of the processes. Such a prior encourages a minimisation of these differences, and can help to ensure that sensible, non-explosive models are estimated, and also help the sampler get past difficult regions of relative flatness in the parameter space due to colinearities between the within and between subject parameters. However if such a prior is too strong it can also induce difficult dependencies in model parameters, and there are a range of models where one may not wish to have such a prior. To place such a prior, the model$stationarymeanprior and model$stationaryvarprior slots can be changed from the default of NA to a numeric vector, representing the standard deviation of a normal distribution for deviations from stationarity in the relevant parameters. The number of elements in the vector correspond to the number of latent processes.

4.8. Individual level analyses

Individual level results can also be considered, as ctsem includes functionality to output prior (based on all prior observations), updated (based on all prior and current observations), and smoothed (based on all observations) expectations and covariances from the Kalman filter, based on specific subjects models. For ease of comparison, expected manifest indicator scores conditional on prior, updated and smoothed states are also included. This approach allows for: predictions regarding individuals states at any point in time, given any values on the time dependent predictors (external inputs such as interventions or events); residual analysis to check for unmodeled dependencies in the data; or simply as a means of visualization, for comprehension and model sanity checking purposes. An example of such is depicted below, where we see observed and estimated scores for a selected subject from our sample. If we wanted to predict unobserved states in the future, we would need only to specify the appropriate timerange (Prediction into earlier times is possible but may make little sense unless the initial states are restricted to stationarity).

```
ctStanKalman(fit, subjects=2, timerange=c(0,40), timestep=.1, plot=TRUE)
```



4.9. Accessing Stan model code

For diagnosing problems or modifying the model in ways not achievable via the ctsem model specification, one can use ctsem to generate the Stan code and then work directly with that, simply by specifying the argument fit=FALSE to the ctStanFit function, and accessing the `stanmodeltext` subobject of the created object. Any altered code can be passed back into ctStanFit by using the stanmodeltext argument, which can be convenient for setting up the data in particular.

4.10. Using Rstan functions

The standard rstan output functions such as summary and extract are also available, and the shinystan package provides an excellent browser based interface. The stan fit object is stored under the $stanfit subobject from the ctStanFit output. The parameters which are likely to be of most interest in the output are prefixed by "hmean_" for hyper (population) mean, "hsd_" for hyper standard deviation, and "tipred_" for time independent predictor. Any hmean parameters are returned in the form used for input - so correlations and standard deviations for any of the covariance related parameters. Subject specific parameters are denoted by the matrix they are from, then the first index represents the subject id, followed by standard matrix notation. For example, the 2nd row and 1st column of the DRIFT matrix for subject 8 is "$DRIFT[8, 2, 1]$". Parameters in such matrices are returned in the form used for internal calculations – that is, variance covariance matrices are returned as such, rather than the lower-triangular standard deviation and correlation matrices required for input. The exception to this are the time independent predictor effects, prefixed with "*tipred_*", for which a linear effect of a change of 1 on the predictor is approximated. So although "tipred_TI1" is only truly linear with respect to internal parameterisations, we approximate the linear effect by averaging the effect of a score of +1 or -1 on the predictor, on the population mean. For any subject that substantially differs from the mean, or simply when precise absolute values of the effects are required (as opposed to general directions), they will need to be calculated manually.

4.11. Oscillating, single subject example - sunspots data

In the following example we fit the sunspots data available within R, which has previously been fit by various authors including Tómasson (2013). We have used the same CARMA(2,1) model and obtained similar estimates – some differences are due to the contrast between Bayes and maximum likelihood, though if desired one could adjust the code to fit using maximum likelihood, as here we have only one subject. There are usually some divergent transitions (indicating a difficulty in the sampling chain and a potential threat to inference) generated in this fit – alternate parameterisations or an increase in the adapt_delta control argument to Stan (which defaults to 0.9 in ctsem, with a maximum of 1, though 1 is not recommended...) may help, though such divergences often appear related to the relatively unimportant T0VAR parameters in any case.

4.12. Population standard deviations - understanding the transforms

Internally, we sample parameters that we will refer to here as the 'raw' parameters – these parameters have no bounds and are drawn from normal distributions. Both population mean (internally: hypermeans) and subject level (internally: indparamsbase) raw parameters are drawn from a normal(0, 1) distribution. Depending on the specific parameter, various transformations may be applied to set appropriate bounds and priors. The population standard deviation (hypersd) for these raw parameters is sampled (by default) from a truncated normal(0, 1) distribution. This distribution can be scaled on a per parameter basis by the sdscale multiplier in the model specification, which defaults to 1. The following script shows a didactic sequence of sampling and transformation for a model with a single parameter, the auto effect of the drift matrix, and 50 subjects. Although we sample the priors here, this is merely to reflect the prior and enable understanding and plotting.

```
#population mean and subject level deviations (pre-transformation)

hypermeans_prior <- rnorm(99999, 0, 1)
hypermeans_post <- -2 #hypothetical sample
```

```
indparamsbase_prior <- rnorm(99999, 0, 1)
indparamsbase_post <- rnorm(50, 0, 1) #hypothetical sample

#population standard deviation prior

hypersd_prior <- rnorm(99999, 0, 1)
hypersd_prior <- hypersd_prior[hypersd_prior > 0] #truncation

#population standard deviation posterior

hypersd_post <- .4 #hypothetical

#population cholesky correlation matrix
#lower triangle sampled from uniform(-1, 1),
#upper triangle fixed to 0,
#diagonal calculated according to hypersd.
hypercorrchol_post <- 1 #because only 1 parameter here...

#population cholesky covariance matrix
#here based on mean of hypersd_post, for convenience...
#in reality would have multiple samples.
hypercovchol <- diag(hypercorrchol_post,1) %*%
  diag(hypersd_post,1) %*% diag(hypercorrchol_post,1)

#subject level parameters
#first compute pre transformation parameters
#then transform appropriately (here according to drift auto effect)
indparams <- hypercovchol %*% indparamsbase_post + hypermeans_post
indparams <- -log(exp(-1.5 * indparams) + 1)

#post transformation population standard deviation
hsd_ourparameter <- abs( #via delta approximation
  (-log(exp(-1.5 * (hypermeans_post + hypersd_post)) + 1) -
   -log(exp(-1.5 * (hypermeans_post - hypersd_post)) + 1) ) / 2)
```

Chapter 5.

Understanding the Time Course of Interventions

How long does a treatment take to have maximum effect? Is the effect then maintained, does it dissipate, or perhaps even reverse? Do certain sorts of people respond faster or stronger than others? Is the treatment more effective in the long run for those that respond quickly? These are the sorts of questions we should be able to answer if we truly understand an intervention and the system we apply it to.

The randomised controlled trial is recognised as something of a gold standard for the analysis of interventions, for good reason. Yet still, when it comes to the direct knowledge of the world such a trial can provide to us, we learn only about what happened to some subjects at a particular moment, or moments, of observation. Going beyond such experiential knowledge in order to produce useful, powerful, and predictive inferences about the world requires some assumptions about regularity and stability of the universe.

For some experimental manipulations it may be sufficient that we learn that there tends to be an effect, while the manipulation is occurring, to subjects like those we observed. In such cases we need only the most basic assumptions of universal regularity – similar antecedents result in similar consequences. However there are many effects which are only of interest because we assume they persist in some sense outside the bounds of our observation window – a treatment for depression would not be very useful if it only improved a persons depression while being evaluated! As such, whether we are explicitly conscious of it or not, we always rely on some model of temporal regularity.

In many cases such a model of temporal regularity may be very simplistic and implicit – something like 'there is some sort of continuity in the observed effects even when we are not observing the subjects'. This stance may be adequate if we know enough about the nature of our system at the time scale we are interested in. For instance, if we assess the effect of a treatment on the level of depression 90 days later, we would also expect to know quite a lot about about the effect of the treatment at 89 and 91 days, even though no observations were made on those days. But what about the effect after 30 days, or 180? Probably, most people would agree that additional observations would be necessary. With multiple observations we could then interpolate between and beyond them, but how should this be done? Simple linear interpolation between the strength of an average treatment effect (across subjects) at various occasions can be adequate for some situations and research questions, but we can also go much further. In this work we endeavour to show that adopting a dynamic systems approach can not only yield improved estimates of the effect of an intervention at unobserved times, but can also help us better understand the nature of the intervention and system more fully, and improve possibilities for personalised treatments.

Specifically, in this paper we adopt a continuous time dynamic modelling approach to the problem, based on linear stochastic differential equations. With this approach, variability within a subject over time is partitioned into: stochastic inputs at the system level (latent process variance), deterministic changes based on earlier states of the system; stochastic inputs at the

measurement level (measurement error); deterministic inputs of unknown origin (i.e., trends); and then finally, deterministic inputs of known cause – an intervention. In broad terms, the approach differs from what could be done using latent growth curves by the inclusion of the system noise and dynamics component. Thus, rather than sketching only a *description* of change over time, to the extent possible the generating *process* is also considered, even if only in very broad terms. This can lead to more informative inferences, dependent of course on the quality of the data, assumptions, and modelling. For an introduction to continuous time models see Voelkle et al. (2012), and for more background see Oud (2002). For an example of higher order modelling applications Voelkle and Oud (2013) details the damped linear oscillator. The text from Gardiner (1985) gives a detailed treatment of stochastic differential equations in general.

Interventions in a system dynamics context are already considered in fields such as pharmacodynamics and pharmacokinetics for drug discovery and testing. There they endeavour to describe processes on the causal path between blood plasma concentration and effect, for multiple subjects (see Danhof, de Lange, Della Pasqua, Ploeger & Voskuyl, 2008; Donnet & Samson, 2013, for general and estimation-focused overviews, respectively). Single-subject analysis, wherein rich temporal data are available, have been more commonly undertaken with such an approach, see for instance the insulin dose and glucose monitoring work of Q. Wang et al. (2014). In the realm of psychology, Boker, Staples and Hu (2016) discusses how certain inputs can lead to changes in the equilibrium of a process.

In the remainder of this work we will first consider why and how various input effects may be modelled, and then consider mediation – how we may be able to use interventions to understand relations between the processes – and individual differences. Throughout the work we will provide examples using the ctsem (Driver et al., 2017) software package for R (R Core Team, 2014)

5.1. Latent dynamic model

Although this equation is comprehensively covered in Chapter 1, in this section we reiterate the fundamental equation governing the process dynamics, along with the description of how we handle input effects.

$$\mathrm{d}\boldsymbol{\eta}(t) = \Big(\mathrm{A}\boldsymbol{\eta}(t) + \mathrm{b} + \mathrm{M}\boldsymbol{\chi}(t)\Big)\mathrm{d}t + \mathrm{G}\mathrm{d}\mathrm{W}(t) \tag{5.1}$$

The time dependent predictors $\boldsymbol{\chi}(t)$ represent exogenous inputs to the system, such as an intervention, that may vary over time and are independent of fluctuations in the system. Equation 5.1 shows a generalized form for time dependent predictors, that could be treated a variety of ways depending on the predictors assumed shape, or time course. We use a simple impulse form shown in Equation 5.2, in which the predictors are treated as impacting the processes only at the instant of an observation occasion u, and the effects then transmit through the system in accordance with A as usual. Such a form has the virtue that many alternative shapes are made possible via augmentation of the system state matrices – this chapter describes many examples of this.

$$\boldsymbol{\chi}(t) = \sum_{u \in \mathrm{U}} \mathrm{x}_u \Delta(t - t_u) \tag{5.2}$$

Here, time dependent predictors $\mathrm{x}_u \in \mathbb{R}^l$ are observed at measurement occasions $u \in \mathrm{U}$. The Dirac delta function $\delta(t - t_u)$ is a generalized function that is ∞ at 0 and 0 elsewhere, yet has an integral of 1 (when 0 is in the range of integration). It is useful to model an impulse to a system, and here is scaled by the vector of time dependent predictors x_u. The effect of

these impulses on processes $\boldsymbol{\eta}(t)$ is then $\mathrm{M} \in \mathbb{R}^{v \times l}$. Put simply, the equation means that at the measurement occasion u a time dependent predictor (e.g., intervention) is observed, the system processes spike upwards or downwards by Mx_u. For a typical intervention that probably only occurs once during the observation window, x_u would then be zero for every observation u except when the intervention occurred, where it could take on a dummy coding value such as one, or could reflect the strength of the intervention. Since M is conceptualized as the effect of instantaneous impulses x, which only occur at occasions U and are not continuously present as for the processes $\boldsymbol{\eta}$, the discrete and continuous time forms are equivalent. This means that the effect of some intervention at measurement occasion $u = 3$ is simply $\mathrm{Mx}_{u=3}$ at the instant of the intervention, and at later measurement occasion $u = 4$, the remaining effect is $\mathrm{A}^*_{\Delta t_{u=4}}\mathrm{Mx}_{u=3}$. If the time interval between occasions 3 and 4 is $\Delta t = 2.30$, using Equation 1.2 this translates to $\mathrm{e}^{\mathrm{A}(2.30)}\mathrm{Mx}_{u=3}$.

5.2. Shapes of input effects

When we speak of the effect of an input, we mean the effect of some observed variable that is *not* possible to think of as a continuously present and fluctuating latent process, but is more plausibly viewed as a specific event at a moment in time. So for instance while a persons' fitness is more easily thought of as constantly present and could thus be modelled as a latent process, an event such as 'going for a run' is probably easier to consider as a single event in time. We do not propose strict guidelines here, it would also be possible to model a 'speed of running' latent process, and it is also clear that all events take some time to unfold. However, we suggest it is both reasonable and simpler to model events occurring over very short time scales (relative to observation intervals) as occurring at a single moment in time. Nevertheless, there may be times when events should be thought of as persisting for some finite span of time, and this may also be approximated using the approaches we will outline.

So, although we speak of an input as occurring only at a single moment in time, the *effects* of such an input on the system of interest will persist for some time (in any system with temporal dependencies), and may exhibit a broad range of shapes. While for the sake of clarity we will discuss the shape of an input effect on a process that is otherwise a flat line, what we really mean by 'shape of the effect' is the *difference* between the expected value of the process conditional on an input, and the expected value of the process without any such input. The shape of the effect could then be relative to a randomly fluctuating process, an oscillation, exponential trend, or what have you. Some examples are shown in Figure 5.1.

5.2.1. Basic impulse effect

As formalised by Equation 5.2, the basic form of the effect, and fundamental building block for more complex effects, is that an input at a singular moment in time causes an impulse in the system at that moment, which then dissipates according to the temporal dependencies (drift matrix) in the system. The effect of such an impulse on a first order, mean-reverting and non-stochastic process (i.e., a flat line with any changes driven by deterministic inputs) is shown in the top left of Figure 5.1. The effect will take on a similar shape for any mean-reverting process, it is just not as easy to see when the process is stochastic and or oscillating.

Effects we might plausibly model in such a way are those where the observed input is expected to have a sudden effect on the system, and that effect dissipates in a similar way to the other, unobserved, inputs on the system that are modelled via the stochastic term. An example of such could be the effect of encountering a friend in the street on ones' mood – mood rapidly rises, and declines back to some baseline in much the same way as would occur for other random mood shifting events throughout the day.

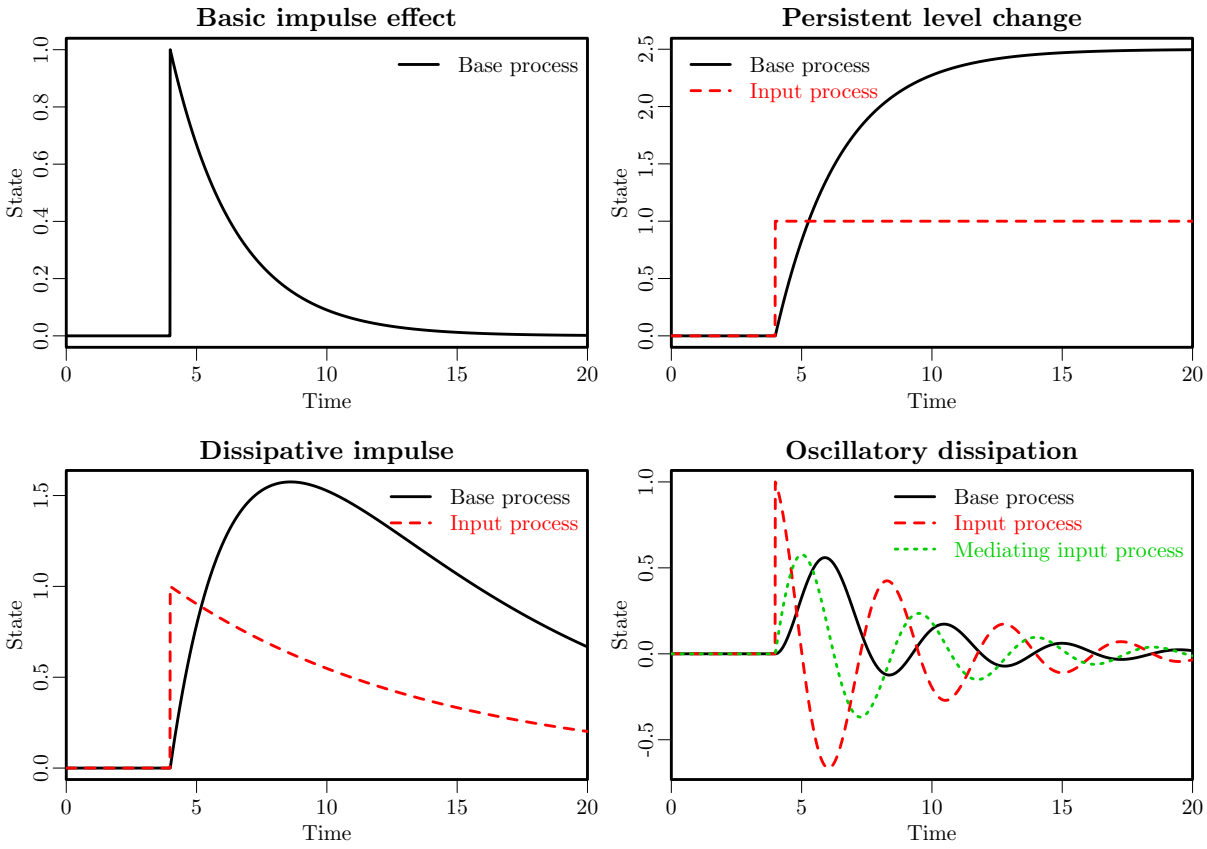


Figure 5.1.: Possible shapes of effects resulting from impulse input affecting a mean-reverting process, given various configurations of the state matrices. For the basic impulse effect no additional processes need to be modelled, while the other examples require either one or two additional processes, that are not directly observed.

Though the equivalence is perhaps not obvious at first, when time-varying covariates are included in a discrete time cross-lagged or latent change score model, and no temporal dependencies in the covariates are modelled, it is just such an impulse effect that is being instantiated. As such, the approaches and thinking we outline in this paper can also be used in the discrete-time case, though care should be taken to consider divergences between the continuous and discrete-time approaches, particularly when models greater than first order (such as those with oscillations) are considered.

An example R script to simulate data from such a model and then fit it, are as follows. Note that for any real world analyses of multiple subjects we would advise to account for between-subject effects (with ctsem either by the MANIFESTTRAITVAR matrix in frequentist configuration, or by allowing individually varying parameters with the Bayesian approach). For the sake of simplicity we ignore such issues for the moment.

```
nlatent=1 #number of latent processes
nmanifest=1 #number of manifest variables
tpoints=30 #number of measurement occasions
ntdpred=1 #number of time dependent predictors
TDPREDMEANS=matrix(0,ntdpred*tpoints,1)
TDPREDMEANS[floor(tpoints/2)]=1 #input after 50% of observations

genm=ctModel(Tpoints=tpoints,
n.latent=nlatent, n.manifest=nmanifest, n.TDpred=ntdpred,
LAMBDA=matrix(c(1), nrow=nmanifest, ncol=nlatent),
DRIFT=matrix(c(-.2),  nrow=nlatent, ncol=nlatent),
DIFFUSION=matrix(c(.1), nrow=nlatent, ncol=nlatent),
MANIFESTVAR=matrix(c(.1), nrow=nmanifest, ncol=nmanifest),
TDPREDEFFECT=matrix(c(1), nrow=nlatent, ncol=ntdpred),
CINT=matrix(c(0), nrow=nlatent, ncol=1),
TDPREDMEANS=TDPREDMEANS,
MANIFESTMEANS=matrix(c(0), nrow=nmanifest, ncol=1))

dat=ctGenerate(ctmodelobj=genm, n.subjects=50, burnin=50)
#ctIndplot(datawide=dat,n.subjects=10,n.manifest=1,Tpoints=tpoints)

fitm=ctModel(Tpoints=tpoints, type="omx",
n.latent=nlatent, n.manifest=nmanifest, n.TDpred=ntdpred,
LAMBDA=matrix(c(1), nrow=nmanifest, ncol=nlatent),
DRIFT=matrix(c("drift11"), nrow=nlatent, ncol=nlatent),
DIFFUSION=matrix(c("diffusion11"), nrow=nlatent, ncol=nlatent),
MANIFESTVAR=matrix(c("merror11"), nrow=nmanifest, ncol=nmanifest),
MANIFESTMEANS=matrix(c("mmeans_Y1"), nrow=nmanifest, ncol=nmanifest),
TDPREDEFFECT=matrix(c("tdpredeffect21"), nrow=nlatent, ncol=ntdpred))

fit=ctFit(dat, fitm)
summary(fit)
```

The matrix forms of the model equations for a basic impulse affecting a first order process are as follows, with underbraced notations denoting the symbol used to represent the matrix in earlier formulas, and where appropriate also the matrix name in the R specification.

$$\underbrace{\mathrm{d}\left[\eta_1\right](t)}_{\mathrm{d}\boldsymbol{\eta}(t)} = \Bigg(\underbrace{\underbrace{\left[\mathrm{drift11}\right]}_{\mathrm{A}}}_{\text{DRIFT}}\underbrace{\left[\eta_1\right](t)}_{\boldsymbol{\eta}(t)} + \underbrace{\underbrace{\left[0\right]}_{\mathrm{b}}}_{\text{CINT}} + \underbrace{\underbrace{\left[\mathrm{tdpredeffect21}\right]}_{\mathrm{M}}}_{\text{TDPREDEFFECT}}\underbrace{\left[\chi_1\right]}_{\boldsymbol{\chi}(t)}\Bigg)dt +$$

$$\underbrace{\underbrace{\left[\mathrm{diffusion11}\right]}_{\mathrm{G}}}_{\text{DIFFUSION}}\underbrace{\mathrm{d}\left[W_1\right](t)}_{\mathrm{dW}(t)}$$

$$\underbrace{\left[Y_1\right](t)}_{\mathrm{Y}(t)} = \underbrace{\underbrace{\left[1\right]}_{\Lambda}}_{\text{LAMBDA}}\underbrace{\left[\eta_1\right](t)}_{\boldsymbol{\eta}(t)} + \underbrace{\underbrace{\left[\mathrm{mmeans\_Y1}\right]}_{\boldsymbol{\tau}}}_{\text{MANIFESTMEANS}} + \underbrace{\left[\epsilon_1\right](t)}_{\boldsymbol{\epsilon}(t)} \quad \text{where } \underbrace{\left[\epsilon_1\right](t)}_{\boldsymbol{\epsilon}(t)} \sim \mathrm{N}\Bigg(\left[0\right], \underbrace{\underbrace{\left[\mathrm{merror11}\right]}_{\Theta}}_{\text{MANIFESTVAR}}\Bigg)$$

5.2.2. Level change effect

In contrast to the impulse effect, some inputs may result in a stable change in the level of a process. Such a change may occur near instantaneously, or more gradually. The top right of Figure 5.1 shows the more gradual change (which in this framework, nevertheless comes about due to an instantaneous change in the unobserved predictor process). We would hope that many treatments may generate such an effect. Consider for instance the effect of an exercise intervention on fitness. In the intervention condition, subjects are encouraged to increase the amount of exercise they do throughout the week. If the intervention is successful, we then wouldn"t necessarily expect to see an immediate improvement in fitness, but would hope that people had begun exercising more in general, which would slowly increase their fitness towards some new level.

There are various ways one could model such an effect. An intuitive approach that may spring to mind would be to code the input variable as 0 for both treatment and control prior to the intervention, then when treatment begins, code the input variable of the treated group as 1 for all further observations. This is somewhat problematic however, as it will result in a series of impulses of the same strength occurring at the observations, with the process declining as normal after an observation, then jumping again with the next impulse. This is not at all representative of a consistent change in level when we consider the process at both observed and unobserved times, i.e. *between the observations*. Nevertheless, if observations are consistently spaced in time, it is potentially adequate for estimating the extent to which the treatment group exhibited a persistent change in their level. For instance, in Figure 5.2, in the example on the left with inconsistent observation intervals, it is clear that both the underlying model and resulting predictions will be very wrong when taking such an input variable coding approach. With the consistent observation time intervals on the right of Figure 5.2 however, we see that although the underlying model is incorrect, predictions at the times when an observation occurs (and the input variable is coded as a 1) do gradually rise towards a new level.

While possible in specific circumstances to model the level change via input variable coding, we would in general argue for a more adequate model specification, which is easily achievable. The approach we will take throughout this work is to model additional, *unobserved*, processes, which have no stochastic component. This might also be referred to as augmenting the state matrices. These additional processes are affected by the impulse of the input variable, and in turn affect our original process of interest, which we will refer to as the *base process*. What have we gained by including such an intermediary? This intermediary, or *predictor process*, is not restricted to an impulse shape (though can assume such a shape), but can exhibit a time

course (or impulse response function) characterised by its‘ own auto effect parameter.

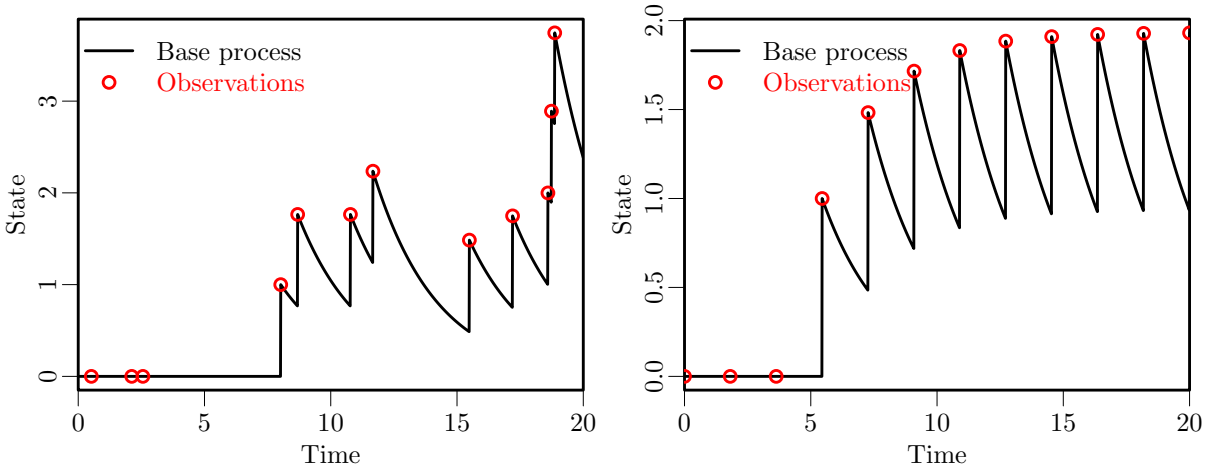


Figure 5.2.: Modelling a persistent level change via input variable coding is strictly incorrect, though with consistent time intervals (or discrete-time models), can sometimes be sufficient – as shown on the right.

One way we might think of such a predictor process is that of directly characterising the effect we are interested in. Consider for instance the effect of room temperature on perspiration, and an experimental situation in which room temperature was changed at some point via a thermostat. We do not have direct experimental control or measurements of temperature, rather, we simply know that the thermostat was changed at some point. So we include ”change of thermostat” as our observed input variable, which generates a sudden and persistent change in one additional predictor process ‘target temperature”, which in turn affects a second predictor process ‘temperature”, which has some effect on attention that we are interested in.

Alternatively, we may have directly observed the manipulation we are interested in, but its‘ effect on the process of interest may occur via some unobserved mediating process. Consider the example of fitness change in response to a cognitive behavioural intervention to motivate exercise. We have a direct observation of ‘intervention took place”, but know that the subjects must actually exercise for any change in fitness to occur. Thus, we could include a dummy coded input variable of ‘intervention”, which generates an instantaneous and persistent change on some unobserved predictor process that we could think of as something like ‘amount of weekly exercise”, which in turn has an effect on our measure of ‘fitness”. Of course, we do not propose any strong interpretation of the unobserved predictor process – it is clear that there will always be many more mediators at many more time scales than we can model. Rather, we propose to model the predictor process sufficiently such that the major features of the causal chain, at the time scale we are interested in, can be adequately captured. So, though we will not necessarily know what the causal links are, with such a model we can aim at least to understand that a) there appears to be some causal chain between our input and our process of interest, and b) in the vicinity of the values of the context we have observed, it exhibits some shape similar to that we have estimated.

Now, how to explicitly formulate such a level change model? Using ctsem, we configure our model just as for the impulse specification, but include an additional latent process. For this process, we fix all input effects that do not stem from the input predictor to 0. These input effects we fix include the stochastic effects of the DIFFUSION matrix as well as intercept elements from the T0MEANS and CINT matrices – these are fixed to 0 because the predictor process should represent only the effect of our observed input variable. Because the process is unobserved, we must identify its‘ scale in some way, which can be done either by fixing the effect of the input variable on the process (To any non-zero value, though 1 would seem

sensible), or by similarly fixing the effect of the predictor process on our process of interest. The following script demonstrates simulating and fitting such a model. Note that because zeroes can cause problems for matrix inversions, where necessary very small deviations from zero are used instead, which has a negligible effect on the model over the time scale of interest.

```
nlatent=2 #number of latent processes
nmanifest=1 #number of manifest variables
tpoints=30 #number of measurement occasions
ntdpred=1 #number of time dependent predictors
TDPREDMEANS=matrix(0,ntdpred*tpoints,1)
TDPREDMEANS[floor(tpoints/2)]=1 #intervention after 50% of observations

genm=ctModel(Tpoints=tpoints,
n.latent=nlatent, n.manifest=nmanifest, n.TDpred=ntdpred,
LAMBDA=matrix(c(1, 0), nrow=nmanifest, ncol=nlatent),
DRIFT=matrix(c(-.2, 0, 1, 0.00001), nrow=nlatent, ncol=nlatent),
DIFFUSION=matrix(c(.1, 0, 0, 0.00001), nrow=nlatent, ncol=nlatent),
MANIFESTVAR=matrix(c(.1), nrow=nmanifest, ncol=nmanifest),
TDPREDEFFECT=matrix(c(0, .2), nrow=nlatent, ncol=ntdpred),
CINT=matrix(c(0, 0), nrow=nlatent, ncol=1),
TDPREDMEANS=TDPREDMEANS,
MANIFESTMEANS=matrix(c(0), nrow=nmanifest, ncol=1))

dat=ctGenerate(ctmodelobj=genm, n.subjects=50, burnin=50)
#ctIndplot(datawide=dat,n.subjects=10,n.manifest=1,Tpoints=tpoints)

fitm=ctModel(Tpoints=tpoints, type="omx",
n.latent=nlatent, n.manifest=nmanifest, n.TDpred=ntdpred,
LAMBDA=matrix(c(1, 0), nrow=nmanifest, ncol=nlatent),
DRIFT=matrix(c("drift11", 0, 1, 0.0001), nrow=nlatent, ncol=nlatent),
DIFFUSION=matrix(c("diffusion11", 0, 0, 0.001), nrow=nlatent, ncol=nlatent),
MANIFESTVAR=matrix(c("merror11"), nrow=nmanifest, ncol=nmanifest),
MANIFESTMEANS=matrix(c("mmeans_Y1"), nrow=nmanifest, ncol=nmanifest),
TDPREDEFFECT=matrix(c(0, "tdpredeffect21"), nrow=nlatent, ncol=ntdpred))

fit=ctFit(dat, fitm)
summary(fit)
```

The matrix forms for a level change intervention affecting a first order process are as follows, with underbraced notations denoting the symbol used to represent the matrix in earlier formulas, and where appropriate also the matrix name in the ctsem specification.

$$\underbrace{\mathrm{d}\begin{bmatrix}\eta_1\\ \eta_2\end{bmatrix}(t)}_{\mathrm{d}\boldsymbol{\eta}(t)} = \left(\underbrace{\begin{bmatrix}\text{drift11} & 1\\ 0 & 1e-04\end{bmatrix}}_{\substack{\mathrm{A}\\ \text{DRIFT}}}\underbrace{\begin{bmatrix}\eta_1\\ \eta_2\end{bmatrix}(t)}_{\boldsymbol{\eta}(t)} + \underbrace{\begin{bmatrix}0\\ 0\end{bmatrix}}_{\substack{\mathrm{b}\\ \text{CINT}}} + \underbrace{\begin{bmatrix}0\\ \text{tdpredeffect21}\end{bmatrix}}_{\substack{\mathrm{M}\\ \text{TDPREDEFFECT}}}\underbrace{[\chi_1]}_{\boldsymbol{\chi}(t)}\right)dt +$$

$$\underbrace{\begin{bmatrix}\text{diffusion11} & 0\\ 0 & 0.001\end{bmatrix}}_{\substack{\mathrm{G}\\ \text{DIFFUSION}}}\underbrace{\mathrm{d}\begin{bmatrix}W_1\\ W_2\end{bmatrix}(t)}_{\mathrm{dW}(t)}$$

$$\underbrace{[Y_1](t)}_{\mathrm{Y}(t)} = \underbrace{\begin{bmatrix}1 & 0\end{bmatrix}}_{\substack{\Lambda\\ \text{LAMBDA}}}\underbrace{\begin{bmatrix}\eta_1\\ \eta_2\end{bmatrix}(t)}_{\boldsymbol{\eta}(t)} + \underbrace{[\text{mmeans\_Y1}]}_{\substack{\boldsymbol{\tau}\\ \text{MANIFESTMEANS}}} + \underbrace{[\epsilon_1](t)}_{\boldsymbol{\epsilon}(t)} \quad \text{where } \underbrace{[\epsilon_1](t)}_{\boldsymbol{\epsilon}(t)} \sim \mathrm{N}\left([0], \underbrace{[\text{merror11}]}_{\substack{\Theta\\ \text{MANIFESTVAR}}}\right)$$

5.3. Additional complexity - dissipation, multiple time-scales and oscillations, trends

So far we have discussed two possible extremes in terms of effect shape, the sudden and singular impulse, and the slower but constant level change. These are both somewhat restrictive in terms of the shape of the effects implied by the model, so in many cases it may be worthwhile to further free the possible shape of effects, either as a comparison for the more restrictive model, or directly as the model of interest.

5.3.1. Dissipation

The most obvious and simplest relaxation is to take the level change model, and free the auto effect (diagonal of the drift matrix) for the predictor process. Then, the extent to which the effect of the input persists is directly estimated, rather than assumed to persist forever. As the auto effect takes on more negative values the input effect approaches the basic impulse type, and as the auto effect nears 0 the resulting predictor process approaches the level change type. Values greater than 0 would suggest that the input effect is compounding on itself with time, generating an explosive process. Such explosive processes tend to be highly unrealistic for forecasting much further into the future, but may be an adequate characterisation over the time range considered. Note that, although with a highly negative auto effect the predictor process approaches the impulse shape, the value of the effect strength parameter will need to be much larger in order to match the impulse form with no mediating input predictor process. This is shown in the top two plots of Figure 5.3, while the bottom two show a slower dissipation on the left, and an explosive effect on the right.

Such a dissipative input is probably a very reasonable starting point for modelling the effect of an intervention intended to have long-term consequences, but where it is unclear if the consequences really do persist forever. Were we to use the simple impulse form, our estimated effect would only represent the magnitude of short term changes, and may not represent the bulk of the effect. With the level-change form of predictor, the estimated effect captures changes that persist for all later observations, and thus if the effect actually declines over time, may underestimate the size of the effect at the short and medium term. Instead of these two extremes, it may instead be the case that the intervention was somewhat successful, with change persisting for some time but slowly dissipating. As the script for simulation and fitting of such a model is very similar to that for the level-change model (with the predictor process auto-effect simply given a parameter name rather than a fixed value of near zero), we do not repeat it here.

5.3.2. Multiple time-scales and oscillations

An additional flexibility that one could also consider is to take either the level-change or dissipation type model specification, and free the direct effect of the input (via the TDPREDEFFECT matrix) on the base process itself. This will then allow for estimating both an instantaneous impulse that dissipates according to the base process, and a more persistent predictor process, allowing for short term effects to differ markedly from longer term, potentially even in the opposite direction. Some real-world examples of such could be the effect of various drugs, for which short-term effects are often very different to longer-term effects, or, at a different time-scale, perhaps the effect of harshly disciplining a child – the child"s behaviour may temporarily improve but with negative consequences later. A mild expansion of this approach could involve the specification of two independent predictor processes, each with a distinct effect on the base process (thus ensuring that dynamics of the base process in general are not confounded with dynamics in response to the intervention). Including parameters to try and tease apart dif-

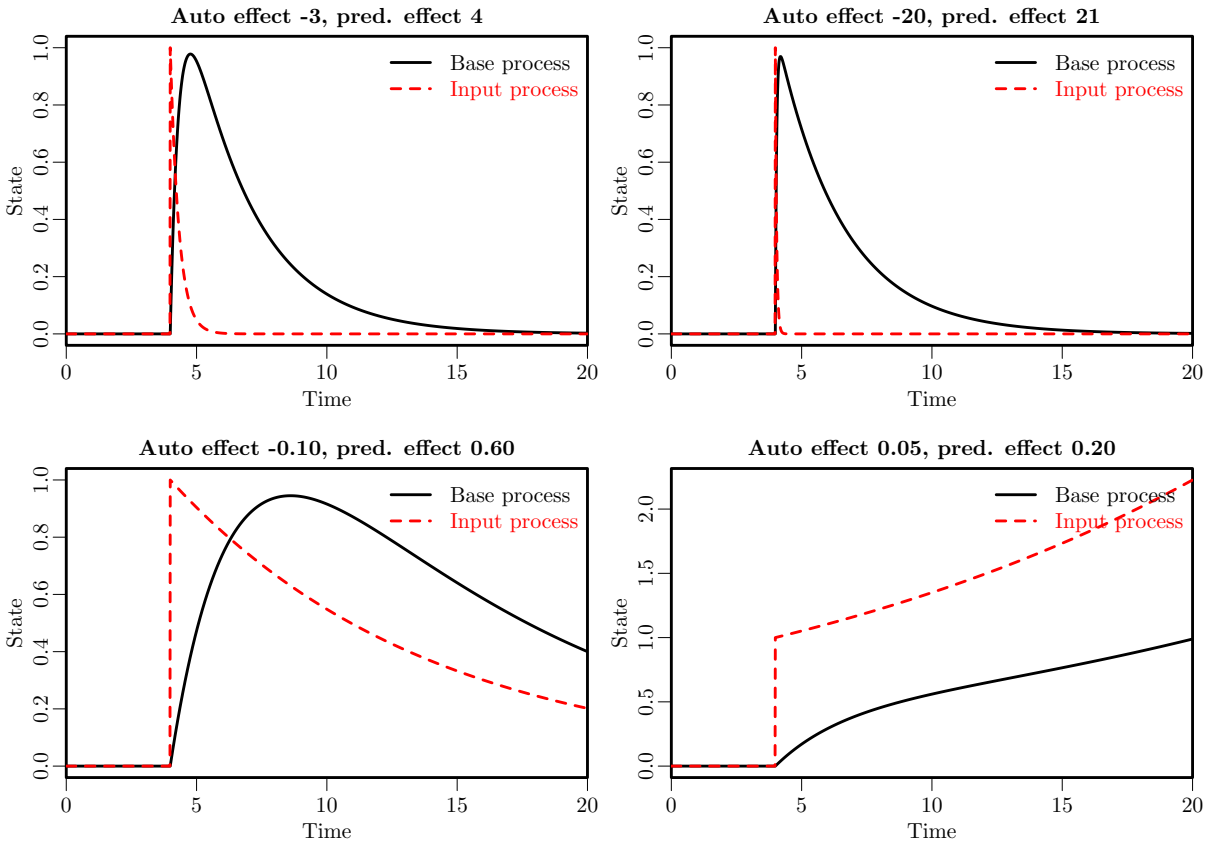


Figure 5.3.: Dissipating input effect, with different parameter values of the predictor process.

ferent time-scale effects will likely make interpretation and hypothesis testing somewhat more complex, and empirical under identification may present an issue for convergence (at least when using the frequentist approach). An approach to mitigate these difficulties may be to revert to the level-change form (wherein the long-term persistence is fixed) rather than estimating the persistence – at least to attain initial estimates.

For more complex shapes of effect due to an intervention, we will need to change from a simple first order predictor process, to higher order configurations. In the vector form used in this work and the relevant estimation software however, higher order processes are anyway always modelled via additional first order processes – so whether one thinks of a single higher order system or multiple first order systems that interact, makes no difference. A damped oscillation resulting from an intervention is then probably the simplest higher-order model, and could be used in similar circumstances to those of the above multiple time-scales example, though the model will have somewhat different properties. In the multiple time-scales model above, a strong interpretation of the model parameters would suggest that there were two distinct and independent effect processes unfolding, one short and one longer term. This is in contrast to the damped oscillation form, in which the longer-term effect unfolds as a deterministic function of the shorter-term effect. We think that for many cases the multiple-time scale approach is probably the more flexible option, though may be more problematic for achieving convergence.

The following R code generates and fits a system with a single first order process of interest (our base process), which is impacted upon by an input effect that generates a dissipating oscillation. Figure 5.4 plots an example of such an input effect, as well as an input effect comprised of two independent predictor processes – one fast and negative, the other slow and positive.

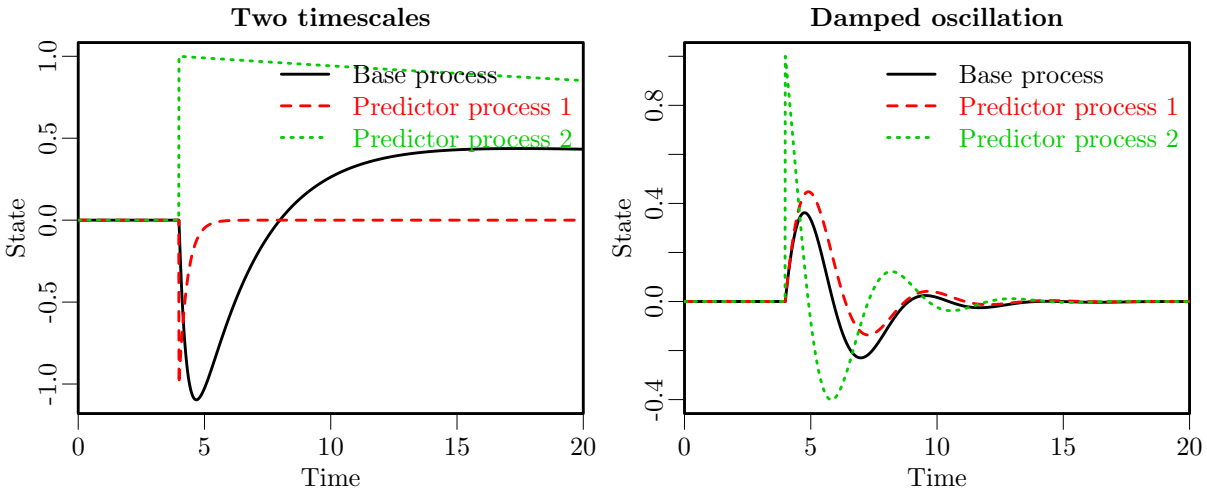


Figure 5.4.: Oscillating input effects. On the left the predictor process is a classic damped linear oscillator, involving two coupled processes. On the right, the two predictor processes are independent, with each having a direct effect on the base process. A simpler variant of the latter may neglect to include the short-term predictor process, instead having the input directly impact the base process.

```
nlatent=3
nmanifest=1
tpoints=30
ntdpred=1
TDPREDMEANS=matrix(0,ntdpred*(tpoints),1)
TDPREDMEANS[floor(tpoints/2)]=1

genm=ctModel(Tpoints=tpoints,
n.latent=nlatent, n.manifest=nmanifest, n.TDpred=ntdpred,
LAMBDA=matrix(c(1, 0, 0), nrow=nmanifest, ncol=nlatent),
DRIFT=matrix(c(-.2, 1, 0,
0, 0,1,
0,-2,-.1), byrow=TRUE, nrow=nlatent, ncol=nlatent),
DIFFUSION=matrix(c(.1, 0, 0,
0,0.0001,0,
0,0,.0001), byrow=TRUE, nrow=nlatent, ncol=nlatent),
MANIFESTVAR=matrix(c(.1), nrow=nmanifest, ncol=nmanifest),
TDPREDEFFECT=matrix(c(0, 0, 1), nrow=nlatent, ncol=ntdpred),
CINT=matrix(c(0), nrow=nlatent, ncol=1),
TDPREDMEANS=TDPREDMEANS,
MANIFESTMEANS=matrix(c(0), nrow=nmanifest, ncol=1))

dat=ctGenerate(ctmodelobj=genm, n.subjects=50, burnin=50)
#ctIndplot(datawide=dat,n.subjects=10,n.manifest=1,Tpoints=tpoints)

fitm=ctModel(Tpoints=tpoints,
n.latent=nlatent, n.manifest=nmanifest, n.TDpred=ntdpred,
LAMBDA=matrix(c(1, 0, 0), nrow=nmanifest, ncol=nlatent),
DRIFT=matrix(c("drift11", 1, 0,
0, 0,1,
0,"drift32","drift33"), byrow=TRUE, nrow=nlatent, ncol=nlatent),
DIFFUSION=matrix(c("diffusion11", 0, 0,
0,0.0001,0,
0,0,.0001), byrow=TRUE, nrow=nlatent, ncol=nlatent),
TDPREDEFFECT=matrix(c(0, 0, "tdpredeffect31"), nrow=nlatent, ncol=ntdpred),
CINT=matrix(c(0), nrow=nlatent, ncol=1),
```

```
TOMEANS=matrix(c("t0mean1",0,0),nrow=nlatent,ncol=1),
MANIFESTVAR=matrix(c("merror11"), nrow=nmanifest, ncol=nmanifest),
MANIFESTMEANS=matrix(c("mmeans_Y1"), nrow=nmanifest, ncol=1))

fit=ctFit(dat, fitm)
summary(fit)
```

The matrix forms for an intervention effect that first rises and then oscillates back to equilibrium, and which affects a first order process, are as follows. Underbraced notations denoting the symbol used to represent the matrix in earlier formulas, and where appropriate also the matrix name in the ctsem specification.

$$\underbrace{\mathrm{d}\begin{bmatrix}\eta_1\\ \eta_2\\ \eta_3\end{bmatrix}(t)}_{\mathrm{d}\boldsymbol{\eta}(t)} = \left(\underbrace{\begin{bmatrix}\text{drift11} & 1 & 0\\ 0 & 0 & 1\\ 0 & \text{drift32} & \text{drift33}\end{bmatrix}}_{\underset{\text{DRIFT}}{\text{A}}}\underbrace{\begin{bmatrix}\eta_1\\ \eta_2\\ \eta_3\end{bmatrix}(t)}_{\boldsymbol{\eta}(t)} + \underbrace{\begin{bmatrix}0\\ 0\\ 0\end{bmatrix}}_{\underset{\text{CINT}}{\text{b}}} + \underbrace{\begin{bmatrix}0\\ 0\\ \text{tdpredeffect31}\end{bmatrix}}_{\underset{\text{TDPREDEFFECT}}{\text{M}}}\underbrace{\begin{bmatrix}\chi_1\end{bmatrix}}_{\boldsymbol{\chi}(t)}\right)dt +$$

$$\underbrace{\begin{bmatrix}\text{diffusion11} & 0 & 0\\ 0 & 1e-04 & 0\\ 0 & 0 & 1e-04\end{bmatrix}}_{\underset{\text{DIFFUSION}}{\text{G}}}\underbrace{\mathrm{d}\begin{bmatrix}W_1\\ W_2\\ W_3\end{bmatrix}(t)}_{\mathrm{dW}(t)}$$

$$\underbrace{\begin{bmatrix}Y_1\end{bmatrix}(t)}_{\mathrm{Y}(t)} = \underbrace{\begin{bmatrix}1 & 0 & 0\end{bmatrix}}_{\underset{\text{LAMBDA}}{\Lambda}}\underbrace{\begin{bmatrix}\eta_1\\ \eta_2\\ \eta_3\end{bmatrix}(t)}_{\boldsymbol{\eta}(t)} + \underbrace{\begin{bmatrix}\text{mmeans\_Y1}\end{bmatrix}}_{\underset{\text{MANIFESTMEANS}}{\boldsymbol{\tau}}} + \underbrace{\begin{bmatrix}\epsilon_1\end{bmatrix}(t)}_{\boldsymbol{\epsilon}(t)} \quad \text{where } \underbrace{\begin{bmatrix}\epsilon_1\end{bmatrix}(t)}_{\boldsymbol{\epsilon}(t)} \sim \mathrm{N}\left(\begin{bmatrix}0\end{bmatrix}, \underbrace{\begin{bmatrix}\text{merror11}\end{bmatrix}}_{\underset{\text{MANIFESTVAR}}{\Theta}}\right)$$

5.3.3. Trends

So far we have been using models where the base process, i.e., fitness, is assumed to be stationary over the time window we are interested in (conditional on any inputs). This means that given knowledge of only the observation times (and not any of the values), our expectations for the unobserved process states will always be the same – neither expectation nor uncertainty regarding our processes is directly dependent on time. While likely unrealistic in a strict sense for all or most cases, so long as prediction errors from time point to time point are not obviously non-Gaussian there may be little reason for concern. However, cases such as long term development, e.g., when observing from childhood to adulthood, are likely to exhibit substantial trends. If unaccounted for such trends are likely to result in highly non-Gaussian prediction errors, violating the assumptions of our model. Furthermore, there may be cases where influencing such a trend via an intervention is of interest, and we thus need to be able to incorporate a long term trend in our model, and include any potential effects of the input on the trend.

Let us consider an example of the influence of a health intervention on reading ability, in children of the developing world. Such a health intervention might consist of a short period involving health checks, treatment, and education, with the idea that better health may facilitate learning, both at school and elsewhere. For an observation window of years with a limited observation frequency, the intervention period can reasonably be treated as a singular event.

To specify such a model, we specify our base process as usual, capturing short-term fluctuations in our process of interest, reading ability. We then need to include an additional

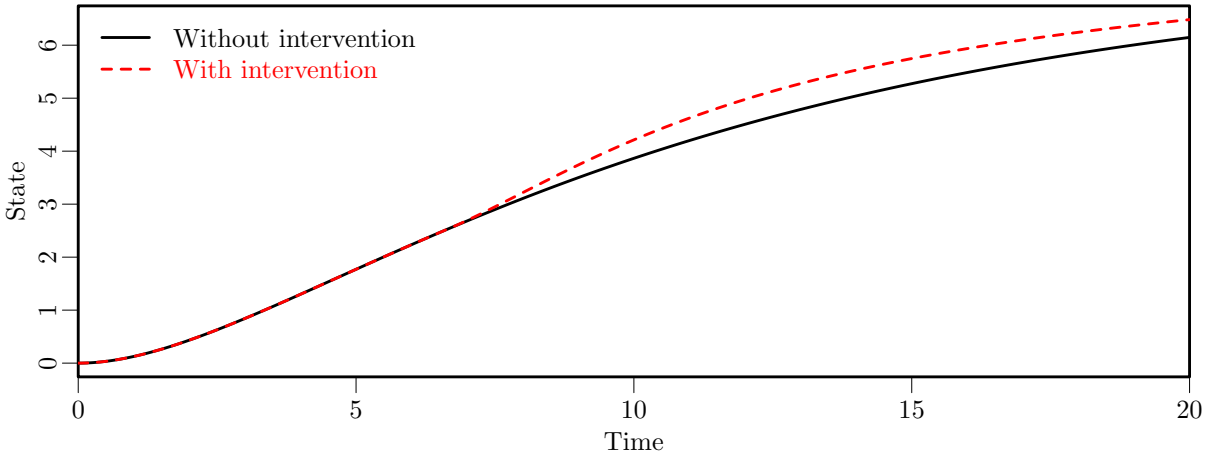


Figure 5.5.: Input effects on a developmental trend. The intervention at time 6 increases the trend slope temporarily.

process that captures the slow trend component. The initial state (T0MEANS), temporal effects between the trend process and other processes, and stochastic input (DIFFUSION) of this trend process are fixed to zero (or near zero), and in contrast to earlier models, we need to include a non-zero continuous-time intercept parameter, to capture the unknown trend size. [1] Other components are estimated as usual. Then we include an input effect onto some form of predictor process, and have this predictor process affect the trend process. In this case, our measurement model must reflect that our measurements are a summation of both base and trend processes.

```
nlatent=3
nmanifest=1
tpoints=30
ntdpred=1
TDPREDMEANS=matrix(0,ntdpred*(tpoints),1)
TDPREDMEANS[floor(tpoints/2)]=1

genm=ctModel(Tpoints=tpoints,
n.latent=nlatent, n.manifest=nmanifest, n.TDpred=ntdpred,
LAMBDA=matrix(c(1, 1, 0), nrow=nmanifest, ncol=nlatent),
DRIFT=matrix(c(-.5, 0, 0,
0, -.05,1,
0,0,-.3), byrow=TRUE, nrow=nlatent, ncol=nlatent),
DIFFUSION=matrix(c(1, 0, 0,
0,0.0001,0,
0,0,.0001), byrow=TRUE, nrow=nlatent, ncol=nlatent),
MANIFESTVAR=matrix(c(.1), nrow=nmanifest, ncol=nmanifest),
TDPREDEFFECT=matrix(c(0, 0, 3), nrow=nlatent, ncol=ntdpred),
CINT=matrix(c(0,3,0), nrow=nlatent, ncol=1),
TDPREDMEANS=TDPREDMEANS,
MANIFESTMEANS=matrix(c(0), nrow=nmanifest, ncol=1))

dat=ctGenerate(ctmodelobj=genm, n.subjects=50, burnin=5)
#ctIndplot(datawide=dat,n.subjects=10,n.manifest=1,Tpoints=tpoints)

fitm=ctModel(Tpoints=tpoints,
n.latent=nlatent, n.manifest=nmanifest, n.TDpred=ntdpred,
```

[1] In a model with between subjects differences in the trend, variability in this parameter can be accommodated via the TRAITVAR matrix (for frequentist ctsem) or by simply setting the parameter to individually varying (in the Bayesian approach).

```
LAMBDA=matrix(c(1, 1, 0), nrow=nmanifest, ncol=nlatent),
DRIFT=matrix(c("drift11", 0, 0,
0, "drift22",1,
0,0,"drift33"), byrow=TRUE, nrow=nlatent, ncol=nlatent),
DIFFUSION=matrix(c("diffusion11", 0, 0,
0,0.0001,0,
0,0,.0001), byrow=TRUE, nrow=nlatent, ncol=nlatent),
TDPREDEFFECT=matrix(c(0, 0, "tdpredeffect31"), nrow=nlatent, ncol=ntdpred),
CINT=matrix(c(0,"cint2",0), nrow=nlatent, ncol=1),
TOMEANS=matrix(c("t0mean1",0,0),nrow=nlatent,ncol=1),
MANIFESTVAR=matrix(c("merror11"), nrow=nmanifest, ncol=nmanifest),
MANIFESTMEANS=matrix(c("mmeans_Y1"), nrow=nmanifest, ncol=1))

fit=ctFit(dat, fitm)
summary(fit)
```

The matrix forms for an intervention effect on the long-term trend of a process are as follows. Underbraced notations denote the symbol used to represent the matrix in earlier formulas, and where appropriate also the matrix name in the ctsem specification.

$$\underbrace{\mathrm{d}\begin{bmatrix}\eta_1\\ \eta_2\\ \eta_3\end{bmatrix}(t)}_{\mathrm{d}\boldsymbol{\eta}(t)} = \Bigg(\underbrace{\begin{bmatrix}\mathrm{drift11} & 0 & 0\\ 0 & \mathrm{drift22} & 1\\ 0 & 0 & \mathrm{drift33}\end{bmatrix}}_{\substack{\mathrm{A}\\ \mathrm{DRIFT}}}\underbrace{\begin{bmatrix}\eta_1\\ \eta_2\\ \eta_3\end{bmatrix}(t)}_{\boldsymbol{\eta}(t)} + \underbrace{\begin{bmatrix}0\\ \mathrm{cint2}\\ 0\end{bmatrix}}_{\substack{\mathrm{b}\\ \mathrm{CINT}}} + \underbrace{\begin{bmatrix}0\\ 0\\ \mathrm{tdpredeffect31}\end{bmatrix}}_{\substack{\mathrm{M}\\ \mathrm{TDPREDEFFECT}}}\underbrace{\begin{bmatrix}\chi_1\end{bmatrix}}_{\boldsymbol{\chi}(t)}\Bigg)dt +$$

$$\underbrace{\begin{bmatrix}\mathrm{diffusion11} & 0 & 0\\ 0 & 1e-04 & 0\\ 0 & 0 & 1e-04\end{bmatrix}}_{\substack{\mathrm{G}\\ \mathrm{DIFFUSION}}}\underbrace{\mathrm{d}\begin{bmatrix}W_1\\ W_2\\ W_3\end{bmatrix}(t)}_{\mathrm{dW}(t)}$$

$$\underbrace{\begin{bmatrix}Y_1\end{bmatrix}(t)}_{\mathrm{Y}(t)} = \underbrace{\begin{bmatrix}1 & 1 & 0\end{bmatrix}}_{\substack{\Lambda\\ \mathrm{LAMBDA}}}\underbrace{\begin{bmatrix}\eta_1\\ \eta_2\\ \eta_3\end{bmatrix}(t)}_{\boldsymbol{\eta}(t)} + \underbrace{\begin{bmatrix}\mathrm{mmeans\_Y1}\end{bmatrix}}_{\substack{\boldsymbol{\tau}\\ \mathrm{MANIFESTMEANS}}} + \underbrace{\begin{bmatrix}\epsilon_1\end{bmatrix}(t)}_{\boldsymbol{\epsilon}(t)} \quad \text{where } \underbrace{\begin{bmatrix}\epsilon_1\end{bmatrix}(t)}_{\boldsymbol{\epsilon}(t)} \sim \mathrm{N}\Bigg(\begin{bmatrix}0\end{bmatrix}, \underbrace{\begin{bmatrix}\mathrm{merror11}\end{bmatrix}}_{\substack{\Theta\\ \mathrm{MANIFESTVAR}}}\Bigg)$$

5.3.4. Mediation

Throughout this work, we have modelled a range of different shapes of input effect by including additional processes in our system model, and these processes have not been directly measured – regression strengths (i.e., elements of the LAMBDA matrix in ctsem) directly from these additional processes to data have been zero. One possible interpretation of these unobserved processes is that they represent some aggregate over all mediating processes that occur between the measured input effect and our measured process of interest. While such processes can simply be left unobserved, understanding the mediators of effects is a common goal of psychological research, and the framework outlined here offers good possibilities for such.

Let us consider again the example of an experimental intervention to improve fitness levels in a group of patients. A successful intervention is unlikely to generate a sudden increase in fitness, rather, it could be expected to gradually rise towards some new level, for which we have already discussed a modelling approach. However, suppose we had also observed some measure of the amount of daily exercise. Conditional on daily exercise, it seems unlikely that the intervention would have any further effect on fitness – daily exercise mediates the effect of the intervention

on fitness. We can test such a theory by comparing a model which includes both mediated and direct effects on fitness, with one that includes only mediated – taking care that we have allowed sufficient flexibility in the model to capture the shape of the effect. The following R script provides an example of such, and compares a model with effects from the input to all processes, to a restricted model where there the input (and any unobserved predictor processes we need to include) only directly affect exercise, which then affects fitness. In order to provide a fair comparison, the full model for fitting contains an unobserved predictor process that is impacted by the input – however the data does not support such a model. This can make attaining convergence somewhat trickier, so in this case we take more care to also specify the initial latent process covariance matrix T0VAR appropriately (instead of the default of all free parameters), and fit the full model using the parameter estimates of the more restricted model as starting values. The mxCompare function from OpenMx is used to compare the two fits, and will show there is no significant difference between the model with the intervention directly affecting fitness, and the restricted model that contains only the intervention effect on exercise rate.

```
nlatent=3
nmanifest=2
tpoints=30
ntdpred=1
TDPREDMEANS=matrix(0,ntdpred*(tpoints),1)
TDPREDMEANS[floor(tpoints/2)]=1

genm=ctModel(Tpoints=tpoints,
n.latent=nlatent, n.manifest=nmanifest, n.TDpred=ntdpred,
LAMBDA=matrix(c(1, 0, 0, 1, 0, 0), nrow=nmanifest, ncol=nlatent),
DRIFT=matrix(c(-.2, .1, 0,
0, -.3,1,
0,0,.0001), byrow=TRUE, nrow=nlatent, ncol=nlatent),
DIFFUSION=matrix(c(.1, 0, 0,
.3,.2,0,
0,0,.0001), byrow=TRUE, nrow=nlatent, ncol=nlatent),
MANIFESTVAR=matrix(c(.1,0,0,.1), nrow=nmanifest, ncol=nmanifest),
TDPREDEFFECT=matrix(c(0, 0, 1), nrow=nlatent, ncol=ntdpred),
CINT=matrix(c(0), nrow=nlatent, ncol=1),
TDPREDMEANS=TDPREDMEANS,
MANIFESTMEANS=matrix(c(0), nrow=nmanifest, ncol=1))

dat=ctGenerate(ctmodelobj=genm, n.subjects=50, burnin=50)
#ctIndplot(datawide=dat,n.subjects=10,n.manifest=2,Tpoints=tpoints)

nlatent=4 #because for our fit we include extra predictor process
fullm=ctModel(Tpoints=tpoints, #type="stanct",
n.latent=nlatent, n.manifest=nmanifest, n.TDpred=ntdpred,
LAMBDA=matrix(c(1, 0, 0, 0, 0, 1, 0, 0), nrow=nmanifest, ncol=nlatent),
DRIFT=matrix(c("drift11", 1, "drift13", 0,
0, "drift22", 0, 0,
0, 0, "drift33", 1,
0, 0, 0, "drift44"), byrow=TRUE, nrow=nlatent, ncol=nlatent),
DIFFUSION=matrix(c("diffusion11", 0, 0, 0,
0, .0001, 0, 0,
"diffusion31", 0, "diffusion33", 0,
0, 0, 0, .0001), byrow=TRUE, nrow=nlatent, ncol=nlatent),
T0VAR=matrix(c("t0var11", 0, 0, 0,
0, .0001, 0, 0,
"t0var31", 0, "t0var33", 0,
0, 0, 0, .0001), byrow=TRUE, nrow=nlatent, ncol=nlatent),
TDPREDEFFECT=matrix(c("tdpredeffect11", "tdpredeffect21",
"tdpredeffect31", "tdpredeffect41"), nrow=nlatent, ncol=ntdpred),
```

```
CINT=matrix(c(0), nrow=nlatent, ncol=1),
T0MEANS=matrix(c("t0mean1", 0, "t0mean3",0),nrow=nlatent,ncol=1),
MANIFESTVAR=matrix(c("merror11",0,0,"merror22"), nrow=nmanifest, ncol=nmanifest),
MANIFESTMEANS=matrix(c("mmeans_fit","mmeans_ex"), nrow=nmanifest, ncol=1))

mediationm=fullm
mediationm$TDPREDEFFECT[1:2,1]=0

mediationfit=ctFit(dat, mediationm)

fullfit=ctFit(dat, fullm,carefulFit=FALSE,
omxStartValues = omxGetParameters(mediationfit$mxobj))

mxCompare(base = fullfit$mxobj, comparison = mediationfit$mxobj)
```

The matrix forms for the full (not yet restricted) mediation model, with an intervention affecting a measured first order process that in turn affects another measured first order process, are as follows. Underbraced notations denote the symbol used to represent the matrix in earlier formulas, and where appropriate also the matrix name in the ctsem specification.

$$\underbrace{\mathrm{d}\begin{bmatrix}\eta_1\\ \eta_2\\ \eta_3\\ \eta_4\end{bmatrix}(t)}_{\mathrm{d}\boldsymbol{\eta}(t)} = \Bigg(\underbrace{\begin{bmatrix}\text{drift11} & 1 & \text{drift13} & 0\\ 0 & \text{drift22} & 0 & 0\\ 0 & 0 & \text{drift33} & 1\\ 0 & 0 & 0 & \text{drift44}\end{bmatrix}}_{\substack{\mathrm{A}\\ \text{DRIFT}}}\underbrace{\begin{bmatrix}\eta_1\\ \eta_2\\ \eta_3\\ \eta_4\end{bmatrix}(t)}_{\boldsymbol{\eta}(t)} + \underbrace{\begin{bmatrix}0\\0\\0\\0\end{bmatrix}}_{\substack{\mathrm{b}\\ \text{CINT}}} + \underbrace{\begin{bmatrix}\text{tdpredeffect11}\\ \text{tdpredeffect21}\\ \text{tdpredeffect31}\\ \text{tdpredeffect41}\end{bmatrix}}_{\substack{\mathrm{M}\\ \text{TDPREDEFFECT}}}\underbrace{[\chi_1]}_{\boldsymbol{\chi}(t)}\Bigg)dt +$$

$$\underbrace{\begin{bmatrix}\text{diffusion11} & 0 & 0 & 0\\ 0 & 1e-04 & 0 & 0\\ \text{diffusion31} & 0 & \text{diffusion33} & 0\\ 0 & 0 & 0 & 1e-04\end{bmatrix}}_{\substack{\mathrm{G}\\ \text{DIFFUSION}}}\underbrace{\mathrm{d}\begin{bmatrix}W_1\\ W_2\\ W_3\\ W_4\end{bmatrix}(t)}_{\mathrm{dW}(t)}$$

$$\underbrace{\begin{bmatrix}Y_1\\ Y_2\end{bmatrix}(t)}_{\mathrm{Y}(t)} = \underbrace{\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 0 & 1 & 0\end{bmatrix}}_{\substack{\Lambda\\ \text{LAMBDA}}}\underbrace{\begin{bmatrix}\eta_1\\ \eta_2\\ \eta_3\\ \eta_4\end{bmatrix}(t)}_{\boldsymbol{\eta}(t)} + \underbrace{\begin{bmatrix}\text{mmeans\_fit}\\ \text{mmeans\_ex}\end{bmatrix}}_{\substack{\boldsymbol{\tau}\\ \text{MANIFESTMEANS}}} + \underbrace{\begin{bmatrix}\epsilon_1\\ \epsilon_2\end{bmatrix}(t)}_{\boldsymbol{\epsilon}(t)}$$

$$\text{where } \underbrace{\begin{bmatrix}\epsilon_1\\ \epsilon_2\end{bmatrix}(t)}_{\boldsymbol{\epsilon}(t)} \sim \mathrm{N}\Bigg(\begin{bmatrix}0\\0\end{bmatrix}, \underbrace{\begin{bmatrix}\text{merror11} & 0\\ 0 & \text{merror22}\end{bmatrix}}_{\substack{\Theta\\ \text{MANIFESTVAR}}}\Bigg)$$

5.3.5. Individual differences

While understanding how the *average* effect of an intervention develops over time is useful, it has long been observed that any such average may not be representative of the development in any single individual – the exercise intervention we have discussed may be more effective for those who live near a park or recreational space, for instance, as the barriers to following the intervention guidelines are lower. An extreme approach to such a problem is to treat individuals as entirely distinct, but this requires very many observations per subject if even moderately

flexible dynamic models are to be fitted, and also raises questions as to how one should treat individuals for which no observations exist. A benefit to this extreme view is that modelling is simplified, and the previously discussed approaches suffice.

A more flexible approach to individual differences is that of random effects (or hierarchical models). These approaches treat individuals as somewhat similar, and estimate the extent of this similarity. This allows for situations where some individuals have been observed many times and others very few, with the resulting model for those observed only few times relying more on the average model across all individuals. For more extended discussion on such models in this context see Driver and Voelkle, In Press. Both frequentist and Bayesian approaches for random (or individual specific) effects of observed input variables on latent processes are relatively straightforward, however random effects on the parameters of any unobserved predictor processes are more complicated in the frequentist case. As such, we demonstrate the case of individual differences using the Bayesian formulation of the ctsem software, which can take longer to fit – in this case roughly 5 minutes on a modern PC.

For this example, we will look how the strength and persistence of an intervention varies in our sample, and relate this variation to an observed covariate. For this we will use the dissipative predictor model developed in Section 5.3.1, but allow for variation in the strength (parameters of the TDPREDEFFECT matrix) and persistence (the drift auto-effect parameter of the unobserved predictor process).

```
nlatent=2 #number of latent processes
nmanifest=1 #number of manifest variables
tpoints=30 #number of measurement occasions
ntdpred=1 #number of time dependent predictors
nsubjects=30 #number of subjects
TDPREDMEANS=matrix(0,ntdpred*tpoints,1)
TDPREDMEANS[floor(tpoints/2)]=1 #intervention after 50% of observations

genm=ctModel(Tpoints=tpoints,
n.latent=nlatent, n.manifest=nmanifest, n.TDpred=ntdpred,
LAMBDA=matrix(c(1, 0), nrow=nmanifest, ncol=nlatent),
DRIFT=matrix(c(-.2, 0, 1, -.2), nrow=nlatent, ncol=nlatent),
DIFFUSION=matrix(c(.1, 0, 0, 0.00001), nrow=nlatent, ncol=nlatent),
MANIFESTVAR=matrix(c(.1), nrow=nmanifest, ncol=nmanifest),
TDPREDEFFECT=matrix(c(0, .2), nrow=nlatent, ncol=ntdpred),
CINT=matrix(c(0, 0), nrow=nlatent, ncol=1),
TDPREDMEANS=TDPREDMEANS,
MANIFESTMEANS=matrix(c(0), nrow=nmanifest, ncol=1))

library(plyr)
dat=aaply(1:nsubjects,1,function(x){ #generate data w random parameter in DRIFT
tempm=genm
stdage=rnorm(1)
tempm$DRIFT[2,2]=-exp(rnorm(1,-2,.5)+stdage*.5)
cbind(ctGenerate(ctmodelobj=tempm, n.subjects=1, burnin=50),stdage)
})
#ctIndplot(datawide=dat,n.subjects=10,n.manifest=1,Tpoints=tpoints)

#convert to long format used by Bayesian ctsem
datlong=ctWideToLong(datawide = dat,Tpoints = tpoints,n.manifest = nmanifest,
n.TDpred = ntdpred,n.TIpred = 1,manifestNames = c("Y1"),
TDpredNames = c("TD1"), TIpredNames=c("stdage"))
datlong=ctDeintervalise(datlong) #convert intervals to abs time

fitm=ctModel(Tpoints=tpoints, type="stanct",
n.latent=nlatent, n.manifest=nmanifest, n.TDpred=ntdpred,
n.TIpred=1, TIpredNames = "stdage",
LAMBDA=matrix(c(1, 0), nrow=nmanifest, ncol=nlatent),
```

```
DRIFT=matrix(c("drift11", 0, 1, "drift22"), nrow=nlatent, ncol=nlatent),
DIFFUSION=matrix(c("diffusion11", 0, 0, 0.001), nrow=nlatent, ncol=nlatent),
MANIFESTVAR=matrix(c("merror11"), nrow=nmanifest, ncol=nmanifest),
MANIFESTMEANS=matrix(c("mmeans_Y1"), nrow=nmanifest, ncol=nmanifest),
TDPREDEFFECT=matrix(c(0, "tdpredeffect21"), nrow=nlatent, ncol=ntdpred))

#only the persistence and strength of the predictor effect varies across individuals
fitm$pars$indvarying[-c(8,18)] = FALSE

#and thus standardised age can only affect those parameters
fitm$pars$stdage_effect[-c(8,18)] = FALSE

fit=ctStanFit(datlong, fitm, iter = 400, chains=3)
summary(fit)
```

Now, the summary of fit gives an estimated population standard deviation of the persistence and strength of the predictor effect (under *$popsd*), and also an estimate of the effect of standardised age on these parameters (under *$tipreds*). In this case there is no genuine effect of age on the effect strength, but it is important to allow for this effect because of the strong dependence between the strength and persistence parameters – it may be difficult to distinguish one rising from the other lowering in some datasets.

5.4. Discussion

The effect of some intervention or event on an ongoing process can manifest in many forms over a range of time scales. We have shown that using a singular impulse (the Dirac delta) as an exogenous input effect in continuous time dynamic models, a wide variety of possible shapes can be estimated by including additional unobserved mediating processes. This allows for changes in subjects‘ baseline levels or trends, that can happen either instantaneously or gradually. Such changes can dissipate very rapidly, persist for the entire observation window, or even build on themselves over time. Changes at different time scales need not even be in the same direction - we have shown how one may model oscillations back to equilibrium, or also an initial change with a slower recovery to a new equilibrium. Such an approach can be used both to formulate and test hypothesis regarding the response over time of individuals to some intervention or stimulus, or in a more exploratory approach using a flexible initial specification or iterative model development. We have demonstrated possibilities of formalising and testing mediation models, as well as for examining relations between individuals‘ specific traits and their response to an intervention – a key component for personalised medicine approaches.

An aspect which has been implicit throughout the work so far is that of causal inference. We do not think it controversial to state that, to the extent that the model is accurately specified and the observed input effects are exogenous to the system processes, causal interpretations of the input effects and their time course are entirely reasonable. While a fully explicated formal treatment of causality is beyond the scope of this work, we will offer some comments on exogeneity of inputs, and accurate model specification in general.

When input effects are not exogenous, as for instance when they are used to model events in a persons‘ life that the person can have some influence over – e.g., ending a relationship – it may still be reasonable to model input effects as discussed, but interpretation is far less clear. In such a case, the observed response to a particular type of event may still be interesting, but the response cannot be assumed to be due to the event specifically – it may instead be due to antecedents that gave rise to the event.

In regard to accurate model specification more generally, we would suggest that a *sufficiently accurate* dynamic model is one that accounts for detectable temporal dynamics and overall trends occurring at the time scales of the observations, in terms of both expectations and

variances. With such a dynamic model, and an adequate measurement model, inferences about the expected time course of input effects, and the relation between processes, can be valid for the time scale and context observed, and then as far as one is willing to generalise this.

Amongst the benefits of the approach we have been discussing, there are also of course some limitations to be considered – these include: The timing of the start of the input effect must be known; the value of the input variables must be known, non-linearities in the input effect are not directly modelled; and the input variable only influences states of the processes, not parameters of the model.

In the approach we have put forward, the timing of all observations, including observations of input variables, are regarded as known, and must be input as data. This could be troublesome if a) the timing of the intervention or event in question is simply not well measured, or b) there is some genuine lag time during which the effect of the input or intervention on the process of interest is truly zero, before suddenly taking effect. In both cases, when the possible error is small relative to the observation time intervals, there are unlikely to be substantial ramifications for model fitting and inference. The latter case may be somewhat trickier to determine, however so far as we can imagine, should be limited to situations involving physical stimuli and very short time scales. For instance, a loud noise must travel from the source to the person, and the person must engage in some low-level processing of the noise, before any startle response would occur. For such situations, if the time-lag is reasonably well known, the timing of the input variable can simply be adjusted. In other cases, extending the model to include a measurement model of the observation timing would seem to make sense, though we have not experimented with such.

Just as we may be uncertain about the exact timing, we may also be uncertain about the exact values of the input variables. This latter case is more straightforward however, simply requiring a measurement model on the input variables. This is not explicitly available within ctsem at time of writing, but could be achieved by treating the input variable as an indicator of a regular process in the system, and fixing the variance and covariance of system noise (Diffusion) for this process to near zero. This configuration would result in a very similar model to the dissipating input effect model we have described, wherein the auto-effect could be freely estimated, or fixed if one wished explicitly for an impulse or level change effect. The strength of the intervention effect would then need to be captured either by the factor loading, or the temporal effect.

Throughout this paper, we have discussed a model in which the input effect is independent of the current system state. So, although the input effect parameters may depend in some way on the stable characteristics of the subject (either their model parameters or included covariates), this effect does not change depending on whether the subject is higher or lower on any processes. This may be problematic when both a) the input effect is actually dependent on the system state, and b) subjects undergo multiple interventions in succession. Repeated interventions may anyway pose additional concerns as a second dose in many cases may not result in exactly the same response as the first. In these situations, it may be wise to include a model wherein a repeated input effect is included as a distinct input variable with its own set of parameters, rather than simply another observation of the same input variable.

Such differential response to repeated inputs is similar to another potential issue, that of non-linear response to dosage. This is a common modelling problem and not specific to this approach. Possible ways of tackling the issue include the addition of quadratic and higher order polynomial versions of the input variables, or binning together ranges of dosage levels and treating these as distinct input variables.

The final limitation we will discuss is that input variables only affect process states, and may not alter the model parameters themselves. While a persistent level change effect is equivalent to a persistent change in the continuous intercept parameter matrix, there are no such analogues for the other parameter matrices such as the temporal dynamics or system

noise. In the case that an intervention substantially changes the dynamics or measurement properties of the system, model parameters estimated using this approach will represent some average over the observation window. There is no cause (that we can see) to expect bias in either state estimates or the expectation given some intervention, but estimates of uncertainty may be inaccurate. In situations where one is concerned about the possibility of a change in the model parameters induced by an intervention, an approach to test for this could be to include a comparison model wherein the parameters are allowed to change in an instantaneous fashion when the intervention occurs, as with interrupted time-series approaches. This is relatively simple to implement, and would correspond to an instantaneous level-change type effect on the model parameters themselves. A more realistic though more complex approach is to include any relevant model parameters in the system state equations, which would require non-linear filtering techniques.

The continuous-time dynamic modelling approach allows for inference with regards to interventions in two important domains. The primary domain is that of the effect of the intervention on any processes of interest – how long do the effects take to manifest, do they vary in direction over time, and are there individual differences in the effects? The second domain is that of the processes themselves – by generating a substantial impulse of known cause, timing and quantity on the processes, this may enable us to better estimate any causal and mediating relations between processes.

So, just how long does a treatment take to reach maximum effect, how long does the effect last, and what sorts of people is it most effective for? By adopting a modelling framework such as we have proposed, and developing an improved understanding of the timing of interventions and their effects, we may be able to better answer such questions.

Chapter 6.

Conclusion

In this dissertation I have described both frequentist and Bayesian approaches to the estimation of hierarchical continuous time dynamic models, developed and described the ctsem software for fitting such models to data, and finished with a discussion and examples for the analysis of interventions in a continuous time dynamic modelling framework.

The basic building block of this work, the stochastic differential equation governing the dynamics of the latent processes (Chapter 1, page 7, Equation 1.1), had already been specified and fit using a structural equation modelling approach by Oud and Jansen (2000). To make the approach more practically applicable, in Chapter 2 I have implemented it in the ctsem software for R, along with various additions to help ensure that sensible and meaningful minima are found by gradient-based optimization, and to ensure that the results are easily interpretable.

The changes to improve estimation include, parameterization changes, stationarity constraints, and various step-wise estimation routines to develop good starting values. The parameterization changes relate to covariance matrices, which are now specified as Cholesky factor covariances with log standard deviations, avoiding the need for parameter boundaries where the optimizers were on occasion getting stuck. The inclusion of stationarity constraints can reduce the number of parameters in the model and help avoid over-fitting, when the implementation is appropriate. With stationarity constraints, rather than allowing variance and mean components of the model to be free at the first observation, transitioning gradually to their long-run stationary values, some or all components can be fixed to the stationary values implied by the main parameters of interest – the drift, diffusion, and continuous time intercept matrices. Because the models can contain many highly dependent parameters, they can be sensitive to starting values, yet good starting values can be quite difficult to determine in advance. To help with this, two routines have been developed to determine starting values. The default routine in the ctsem software is a penalised approach, in which model fit is penalised as certain parameters tend towards extremes. The second approach is an automated step-wise freeing of parameters, wherein the user passes in the full model, but first a series of more restrictive models are fit, and then the full model is fit with starting values based on the restricted fit.

To ease interpretation, parameter matrices are (when appropriate) reported as regular variance and covariance parameters, standardised matrices are included, and the discrete time effects resulting from the model can be plotted over time. With this structural equation implementation of continuous time dynamic modelling, between subjects differences in average levels and overall trends can be accounted for, such that the estimated dynamic model represents an average model for within-subject dynamics across people. This approach allows for relatively fast fitting, and can be used to address questions regarding average within-person dynamics, such as 'do changes in health satisfaction typically predict later changes in life satisfaction, after controlling for common causes' or 'how long, on average, does the intervention take to have maximum effect'.

To allow for questions regarding individual differences in dynamics to be addressed, without the high data requirements of single subject time series approaches, in Chapter 3 I describe a hierarchical Bayesian approach to estimating the continuous time dynamic model. With

such a formulation, all parameters of the dynamic and measurement models can be random across subjects. Such an approach bridges the divide between person-specific, single subject time series approaches, and classical panel data style analyses. Subjects can have their own specific set of parameters for their dynamic and measurement models, yet estimates of those parameters incorporates and benefits from what we know of other subjects. Again of course such an approach in general is by no means new, as both discrete and continuous time dynamic models have been developed with random parameters – however the approach posed in this work resolves some of the limitations of earlier work, such as the symmetric temporal effects of the continuous time approach of Oravecz et al. (2016). To demonstrate the Bayesian approach I showed that we can estimate individual specific models for subjects in long term panel studies such as the GSOEP. Such studies may contain information over a very long amount of time, which is great in terms of investigating developmental questions, but the data are usually not dense enough to estimate individual specific parameters – leveraging information from other individuals makes this possible. Amongst other things, from this analysis we found some evidence that after controlling for common causes, changes in health satisfaction predict later changes in overall satisfaction, and that this effect is stronger for those with generally lower levels of satisfaction.

To check the performance of the Bayesian specification, Chapter 3 also contains a small simulation study. This study showed that when the true mixed-effects model is fit, and the data is sufficiently informative (i.e., large enough N and T), Bayesian and frequentist approaches are comparable – just as would be hoped. For lower N and T, the Bayesian approach is more accurate, has a better match between empirical coverage rates and target coverage rates, but is also more biased. It is typical that accuracy can be improved (variance reduced) at the expense of some bias, the benefit of such a simulation is that the trade-off is quantified. In the illustrated case of a bivariate first order latent process, with moderate auto effects and some cross effect, inferential properties in the lower N and T conditions (N=50 and T=10) seem clearly better in the Bayesian case, with much improved accuracy and coverage rates of interval estimates more closely matching the targeted 95% rate. When the population model is unknown, a conservative approach will typically avoid placing strong restrictions between subjects. When fitting the bivariate model with random effects across all subject level parameters (as only the Bayesian implementation can), performance of course degraded somewhat from that of the true model, but was in general still reasonable – accuracy was still better than with the true model and the frequentist approach, biases were worsened, and coverage rates were still adequate. While these simulation results demonstrate the viability of the approach, the ideal specification of priors is far from certain, and more comprehensive simulation studies are needed to both improve, and to check the limits of, the proposed approach.

To ensure that the hierarchical Bayesian model is useable for applied researchers, it has been incorporated into the ctsem software. Though the usage of the software is similar to the frequentist, SEM approach, there are some important differences, particularly in relation to parameter transformations, covariate effects, and interpretation of results. Chapter 4 gave an overview of using ctsem for the hierarchical Bayesian specification, and the function help files provide more specific detail.

In Chapter 5, I've shown how the model and software described in earlier chapters can be used to improve our understanding of the time course of interventions. Much work in psychology is dedicated to showing that some effect 'exists' or does not. The approach outlined aims instead to understand how an effect arises and changes once some intervention, some experimental input, is given, and is flexible enough to accommodate many possible shapes, as well as individual differences and covariates. A more nuanced understanding of how effects manifest through time could be very beneficial, both in enhancing our capacity to tailor treatments to individuals, and for understanding the causal and mediation processes underlying observed effects. A particular benefit of addressing such questions from a continuous time framework, is that we can be much

more free when collecting data about interventions and processes that are not well understood – if we don't know whether an effect will manifest most strongly after one week or ten, and whether this depends on subject characteristics, we can simply vary measurement intervals in the study. So, one subject may be measured at weeks 2, 3, 5 and 7, while another may be measured at weeks 6, 13, 15, 19 – each providing some portion of the overall picture regarding the time course of effects.

Of course, while the continuous time and hierarchical Bayesian elements of this work allow for considerable flexibility in the model specification, there are limitations to be aware of: Although parameter differences *between* individuals are fully accounted for, for the most part we do not account for parameter variability *within* individuals. Smooth changes in the process means can be accounted for through augmentation of the state matrices (Delsing & Oud, 2008; Oud & Jansen, 2000), and sudden changes can be accounted for using time dependent predictors, but changes in dynamic relationships, or randomly fluctuating parameters, are at present not accounted for. Such effects generally necessitate non-linear stochastic differential equations, and alternative, more complex approaches to dynamic modelling are necessary, as for example in Lu et al. (2015).

In the form described, the model and software is also limited to a linear measurement model with normally distributed errors. However, an option to avoid the Kalman filter and explicitly sample latent states is included in the ctsem software, so although most alternate measurement models are not explicitly specifiable via the ctsem functions at present, non-linearities in the measurement model and link are possible with some modifications to the Stan model that is output by the software.

Additional limitations are those typically found when dealing with either complex hierarchical models, dynamic models, or specifically continuous time dynamic models. These include computation time, parameter dependencies, and parameter interpretation.

Computation time is generally high for hierarchical time series models estimated with sampling approaches typical to Bayesian modelling In this case the continuous time aspect adds some additional computations for every measurement occasion where the time interval changes. Based on experiences so far, using ctsem and Stan in their present forms on a modern desktop PC requires anywhere from a few minutes for simple univariate process models with limited between-subject random effects, to days for complex multi-process multi-measurement models with many subjects and covariates. The satisfaction example from Chapter 3 took approximately 6 hours, for instance. Although performance of the software is at present functional, in that typical sorts of data and models can be fit within minutes or hours, it is nevertheless something of a limitation – running a variety of different models, using large data sets, or specifying models with many latent processes, are all slow enough to pose significant problems to work flow. While on the one hand we may hope for improvements to sampling and other algorithms to gradually reduce these problems, the application of various Bayesian approximation approaches to this problem could also be practically very useful. There is likely also scope within the model parameterization for improvement, as while the current approach incorporates many enhancements suggested in Stan Development Team (2016b), there are many ways of achieving a similar model that have not been tested.

Parameter dependencies in dynamic models pose difficulties both during estimation, and during interpretation. While on the one hand it would be great to specify the model using parameters that were entirely independent of each other, this is not always possible for every parameter, and even when it is, may limit the initial specification and or compound difficulties with interpretation. For instance, rather than parameterizing the innovation covariance matrix using the standard deviations and correlation of the continuous-time diffusion matrix, one alternative we have explored is to estimate directly the asymptotic innovation covariance matrix, $Q^*_{\Delta t=\infty}$. This is beneficial in that the covariance matrix parameters are made independent of the temporal dynamics parameters, and in this case we think also assists in interpretation. Unfortu-

nately however, it limits the possibility to specify more complex dynamics where many elements of the continuous time diffusion matrix may need to be fixed to zero, but the asymptotic latent variance matrix cannot be determined in advance. While the specific parameterizations we propose may need to be adapted, in this work I have aimed for a middle ground approach, trying to balance competing factors for typical use cases.

Interpretation of the continuous time dynamic parameters is typically less intuitive for the applied researcher than the related discrete time parameters. While I would argue that direct interpretation of the latent intercept or variance parameters in either case is problematic, due to the aforementioned parameter dependencies (which one can avoid by transforming any output parameters to the asymptotic, $\Delta t \to \infty$, forms), the need to matrix exponentiate the temporal effects matrix is evidently less intuitive, for anyone thinking in terms of regression effects given some specific time interval. I hope explanations of the drift matrix, and the benefits described encourage people to overcome this. I also hope to encourage better intuition for dynamic models in general, by plotting expected effects (as per Figure 3.4) over a range of time intervals – whether or not a model is fit with a continuous time approach, we typically assume that what we are measuring exists between observations!

While neither hierarchical models nor continuous time dynamic models are themselves new, there have been few approaches put forward combining the two. I believe it is important to describe such models and provide the means to estimate them, because dynamic models offer excellent scope for relating changes in behaviour and cognition over time, and because accurate estimates for single subject dynamic models can require very large numbers of time points. I have highlighted potential applications, by showing that we can estimate individual specific models for subjects in long term panel studies such as the GSOEP. Such studies may contain information over a very long amount of time, which is great in terms of investigating developmental questions, but the data are often not dense enough to estimate individual specific models without the help of information from other individuals. Amongst other things, from the GSOEP analysis there was some evidence that changes in health satisfaction predict later changes in overall satisfaction, and that this effect is stronger for those with generally lower levels of overall and health satisfaction. Adopting a continuous time approach to dynamic modelling also allows for a lot of flexibility for gathering and working with data – subjects may have been measured in previous or ongoing studies, some subjects may be measured many times with small time intervals between measures, or some may be measured only a few times at highly variable, or sparse, intervals. Given that we still do not well understand the time course of many psychological constructs of interest, incorporating such flexibility may be very helpful. I hope that this work offers some means for improving understanding of individuals psychological processes, as well as those factors that relate to differences in such processes between individuals.

Bibliography

Arnold, L. (1974). *Stochastic Differential Equations: Theory and Applications*. New York: John Wiley & Sons.

Balestra, P. & Nerlove, M. (1966, July 1). Pooling cross section and time series data in the estimation of a dynamic model: The demand for natural gas. *Econometrica*, *34*(3), 585–612. doi:10.2307/1909771. JSTOR: 1909771

Barnard, J., McCulloch, R. & Meng, X.-L. (2000). Modeling covariance matrices in terms of standard deviations and correlations, with application to shrinkage. *Statistica Sinica*, 1281–1311. JSTOR: 24306780

Bernardo, J. M. (1979). Reference posterior distributions for Bayesian inference. *Journal of the Royal Statistical Society. Series B (Methodological)*, 113–147. JSTOR: 2985028

Bernardo, J. M., Bayarri, M. J., Berger, J. O., Dawid, A. P., Heckerman, D., Smith, A. F. M. & West, M. (2003). Non-centered parameterisations for hierarchical models and data augmentation. *Bayesian Statistics 7: Proceedings of the Seventh Valencia International Meeting*, 307.

Betancourt, M. J. & Girolami, M. (2013, December 3). Hamiltonian Monte Carlo for hierarchical models. arXiv: 1312.0906 [stat]. Retrieved May 2, 2016, from http://arxiv.org/abs/1312.0906

Boker, S. M. (2001). Differential structural equation modeling of intraindividual variability. *New methods for the analysis of change*, 5–27. doi:10.1037/10409-001

Boker, S. M., Deboeck, P. R., Edler, C. & Keel, P. K. (2010). Generalized local linear approximation of derivatives from time series. In S. -M, E. Ferrer & F. Hsieh (Eds.), *Statistical methods for modeling human dynamics: An interdisciplinary dialogue* (pp. 161–178). The Notre Dame series on quantitative methodology. New York, NY, US: Routledge/Taylor & Francis Group.

Boker, S. M., Staples, A. D. & Hu, Y. (2016). Dynamics of Change and Change in Dynamics. *Journal of Person-Oriented Research*, *2*(1-2), 34–55. doi:10.17505/jpor.2016.05

Bollen, K. A. & Brand, J. E. (2010). A General Panel Model with Random and Fixed Effects: A Structural Equations Approach. *Social Forces*, *89*(1), 1–34. doi:10.1353/sof.2010.0072

Boukhetala, K. & Guidoum, A. C. (2014). Sim. diffproc: Simulation of Diffusion Processes.

Boulton, A. J. (2014, August 31). Bayesian estimation of a continuous-time model for discretely-observed panel data. Retrieved June 28, 2016, from https://kuscholarworks.ku.edu/handle/1808/16843

Box, G. E., Jenkins, G. M., Reinsel, G. C. & Ljung, G. M. (2015). *Time series analysis: Forecasting and control* (5th ed.). John Wiley & Sons.

Bringmann, L. F., Vissers, N., Wichers, M., Geschwind, N., Kuppens, P., Peeters, F., ... Tuerlinckx, F. (2013, April 4). A network approach to psychopathology: New insights into clinical longitudinal data. *PLOS ONE*, *8*(4), e60188. doi:10.1371/journal.pone.0060188

Brouste, A., Fukasawa, M., Hino, H., Iacus, S. M. [Stefano M.], Kamatani, K., Koike, Y., ... Shimuzu, Y. et al. (2014). The yuima Project: A Computational Framework for Simulation and Inference of Stochastic Differential Equations. *Journal of Statistical Software*, *57*(4), 1–51.

Busseri, M. A. & Sadava, S. W. (2011, August). A review of the tripartite structure of subjective well-being: Implications for conceptualization, operationalization, analysis, and synthesis. *Personality and Social Psychology Review*, *15*(3), 290–314. WOS:000292207700003. doi:10.1177/1088868310391271

Carpenter, B., Gelman, A., Hoffman, M. D., Lee, D., Goodrich, B., Betancourt, M., ... Riddell, A. (2017). Stan: A probabilistic programming language. *Journal of Statistical Software*, *76*(1). doi:10.18637/jss.v076.i01

Cattell, R. B. (1963). The structuring of change by P-technique and incremental R-technique. *Problems in measuring change*, 167–198.

Chow, S.-M., Ho, M.-h. R., Hamaker, E. L. & Dolan, C. V. (2010). Equivalence and Differences Between Structural Equation Modeling and State-Space Modeling Techniques. *Structural Equation Modeling: A Multidisciplinary Journal*, *17*(2), 303–332. doi:10.1080/10705511003661553

Chow, S.-M., Lu, Z., Sherwood, A. & Zhu, H. (2014, November 22). Fitting nonlinear ordinary differential equation models with random effects and unknown initial conditions using the stochastic approximation expectation–maximization (SAEM) algorithm. *Psychometrika*, *81*(1), 102–134. doi:10.1007/s11336-014-9431-z

Chow, S.-M., Ram, N., Boker, S. M., Fujita, F. & Clore, G. (2005). Emotion as a thermostat: Representing emotion regulation using a damped oscillator model. *Emotion*, *5*(2), 208–225. doi:10.1037/1528-3542.5.2.208

Danhof, M., de Lange, E. C. M., Della Pasqua, O. E., Ploeger, B. A. & Voskuyl, R. A. (2008). Mechanism-based pharmacokinetic-pharmacodynamic (PK-PD) modeling in translational drug research. *Trends in Pharmacological Sciences*, *29*(4), 186–191. doi:10.1016/j.tips.2008.01.007

Deboeck, P. R., Nicholson, J. S., Bergeman, C. S. & Preacher, K. J. (2013). From modeling long-term growth to short-term fluctuations: Differential equation modeling is the language of change. In R. E. Millsap, L. A. van der Ark, D. M. Bolt & C. M. Woods (Eds.), *New Developments in Quantitative Psychology* (66, pp. 427–447). Springer Proceedings in Mathematics & Statistics. Springer New York. doi:10.1007/978-1-4614-9348-8_28

Delattre, M., Genon-Catalot, V. & Samson, A. (2013, June 1). Maximum likelihood estimation for stochastic differential equations with random effects. *Scandinavian Journal of Statistics*, *40*(2), 322–343. doi:10.1111/j.1467-9469.2012.00813.x

Delsing, M. J. M. H. & Oud, J. H. L. (2008). Analyzing Reciprocal Relationships by Means of the Continuous-Time Autoregressive Latent Trajectory Model. *Statistica Neerlandica*, *62*(1), 58–82. doi:10.1111/j.1467-9574.2007.00386.x

Donnet, S. & Samson, A. (2013). A review on estimation of stochastic differential equations for pharmacokinetic/pharmacodynamic models. *Advanced Drug Delivery Reviews*. Mathematical modeling of systems pharmacogenomics towards personalized drug delivery, *65*(7), 929–939. doi:10.1016/j.addr.2013.03.005

Driver, C. C., Oud, J. H. L. & Voelkle, M. C. (2017). Continuous time structural equation modeling with r package ctsem. *Journal of Statistical Software*, *77*(5). doi:10.18637/jss.v077.i05

Driver, C. C. & Voelkle, M. C. (In Press). Hierarchical Bayesian continuous time dynamic modeling. *Psychological Methods*. Retrieved from https://www.researchgate.net/publication/310747801_Hierarchical_Bayesian_Continuous_Time_Dynamic_Modeling

Durbin, J. & Koopman, S. J. (2012). *Time Series Analysis by State Space Methods: Second Edition*. Oxford University Press.

Evans, M. A., Kim, H.-M. & O'Brien, T. E. (1996). An application of profile-likelihood based confidence interval to capture: Recapture estimators. *Journal of Agricultural, Biological, and Environmental Statistics*, *1*(1), 131–140. doi:10.2307/1400565. JSTOR: 1400565

Finkel, S. E. (1995). *Causal analysis with panel data*. Sage.

Gardiner, C. W. (1985). *Handbook of stochastic methods*. Springer Berlin. Retrieved June 30, 2016, from http://tocs.ulb.tu-darmstadt.de/12326852.pdf

Gasimova, F., Robitzsch, A., Wilhelm, O., Boker, S. M., Hu, Y. & Hülür, G. (2014). Dynamical systems analysis applied to working memory data. *Quantitative Psychology and Measurement*, *5*, 687. doi:10.3389/fpsyg.2014.00687

Gelman, A. (2006, September). Prior distributions for variance parameters in hierarchical models (comment on article by Browne and Draper). *Bayesian Analysis*, *1*(3), 515–534. doi:10.1214/06-BA117A

Gelman, A. & Carlin, J. (2014). Beyond Power Calculations. *Perspectives on Psychological Science*, *9*(6), 641–651. doi:10.1177/1745691614551642. pmid: 26186114

Gelman, A. & Rubin, D. B. (1992). Inference from iterative simulation using multiple sequences. *Statistical Science*, *7*(4), 457–472. doi:10.1214/ss/1177011136. JSTOR: 2246093

Genz, A. & Bretz, F. (2009). *Computation of multivariate normal and t probabilities*. Lecture Notes in Statistics. Berlin, Heidelberg: Springer Berlin Heidelberg. Retrieved August 18, 2016, from http://link.springer.com/10.1007/978-3-642-01689-9

Gill, P. E., Murray, W., Saunders, M. A. & Wright, M. H. (1986). *User's guide for NPSOL (version 4.0): A Fortran package for nonlinear programming*. DTIC Document. Retrieved March 20, 2017, from http://oai.dtic.mil/oai/oai?verb=getRecord&metadataPrefix=html&identifier=ADA169115

Grasman, J., Grasman, R. P. P. P. & van der Maas, H. L. J. (2016, June 28). The dynamics of addiction: Craving versus self-control. *PLOS ONE*, *11*(6). doi:10.1371/journal.pone.0158323

Grewal, M. S. & Andrews, A. P. (2010). Applications of Kalman Filtering in Aerospace 1960 to the Present [Historical Perspectives]. *IEEE Control Systems*, *30*(3), 69–78. 00093. doi:10.1109/MCS.2010.936465

Halaby, C. N. (2004, January 1). Panel models in sociological research: Theory into practice. *Annual Review of Sociology*, *30*(1), 507–544. doi:10.1146/annurev.soc.30.012703.110629

Hamaker, E. L., Kuiper, R. M. & Grasman, R. P. (2015). A critique of the cross-lagged panel model. *Psychological methods*, *20*(1), 102. doi:10.1037/a0038889

Hamaker, E. L., Zhang, Z. & van der Maas, H. L. J. (2009, February 25). Using threshold autoregressive models to study dyadic interactions. *Psychometrika*, *74*(4), 727. doi:10.1007/s11336-009-9113-4

Hamilton, J. (1986). *State-space models*. Elsevier. Retrieved March 20, 2017, from http://econpapers.repec.org/bookchap/eeeecochp/4-50.htm

Headey, B. & Muffels, R. (2014). Trajectories of life satisfaction: Positive feedback loops may explain why life satisfaction changes in multi-year waves, rather than oscillating around a set-point. Retrieved March 21, 2017, from https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2470527

Hertzog, C. & Nesselroade, J. R. (2003). Assessing psychological change in adulthood: An overview of methodological issues. *Psychology and aging*, *18*(4), 639. doi:10.1037/0882-7974.18.4.639

Higham, N. (2009, November 4). The scaling and squaring method for the matrix exponential revisited. *SIAM Review*, *51*(4), 747–764. doi:10.1137/090768539

Homan, M. D. & Gelman, A. (2014, January). The no-u-turn sampler: Adaptively setting path lengths in Hamiltonian Monte Carlo. *J. Mach. Learn. Res. 15*(1), 1593–1623. Retrieved May 2, 2016, from http://dl.acm.org/citation.cfm?id=2627435.2638586

Hunter, M. D. (2014). *State Space Dynamic Mixture Modeling: Finding People With Similar Patterns Of Change* (Doctoral dissertation, University of Oklahoma, Norman, OK).

Iacus, S. M. [Stefano Maria]. (2015). Sde: Simulation and Inference for Stochastic Differential Equations.

Jazwinski, A. H. (2007). *Stochastic processes and filtering theory*. Courier Corporation.

Kalman, R. E. & Bucy, R. S. (1961, March 1). New results in linear filtering and prediction theory. *Journal of Basic Engineering*, *83*(1), 95–108. doi:10.1115/1.3658902

King, A. A., Ionides, E. L., Bretó, C. M., Ellner, S., Kendall, B., Wearing, H., ... Reuman, D. C. (2010). Pomp: Statistical Inference for Partially Observed Markov Processes (r Package). *URL http://pomp. r-forge. r-rproject. org*.

Koval, P., Sütterlin, S. & Kuppens, P. (2016, January 8). Emotional inertia is associated with lower well-being when controlling for differences in emotional context. *Frontiers in Psychology*, *6*. doi:10.3389/fpsyg.2015.01997. pmid: 26779099

Leander, J., Almquist, J., Ahlström, C., Gabrielsson, J. & Jirstrand, M. (2015, February 19). Mixed effects modeling using stochastic differential equations: Illustrated by pharmacokinetic data of nicotinic acid in obese zucker rats. *The AAPS Journal*, *17*(3), 586–596. doi:10.1208/s12248-015-9718-8. pmid: 25693487

Lewandowski, D., Kurowicka, D. & Joe, H. (2009, October). Generating random correlation matrices based on vines and extended onion method. *Journal of Multivariate Analysis*, *100*(9), 1989–2001. doi:10.1016/j.jmva.2009.04.008

Liu, S. (2017, February 1). Person-specific versus multilevel autoregressive models: Accuracy in parameter estimates at the population and individual levels. *British Journal of Mathematical and Statistical Psychology*, n/a–n/a. doi:10.1111/bmsp.12096

Lodewyckx, T., Tuerlinckx, F., Kuppens, P., Allen, N. B. & Sheeber, L. (2011, February). A hierarchical state space approach to affective dynamics. *Journal of Mathematical Psychology*. Special Issue on Hierarchical Bayesian Models, *55*(1), 68–83. doi:10.1016/j.jmp.2010.08.004

Lu, Z.-H., Chow, S.-M., Sherwood, A. & Zhu, H. (2015, September). Bayesian analysis of ambulatory blood pressure dynamics with application to irregularly spaced sparse data. *The Annals of Applied Statistics*, *9*(3), 1601–1620. doi:10.1214/15-AOAS846

Marriott, F. H. C. & Pope, J. A. (1954). Bias in the Estimation of Autocorrelations. *Biometrika*, *41*, 390–402. doi:10.2307/2332719. JSTOR: 2332719

McArdle, J. J. [J. Jack] & McDonald, R. P. (1984). Some algebraic properties of the reticular action model for moment structures. *British Journal of Mathematical and Statistical Psychology*, *37*(2), 234–251. doi:10.1111/j.2044-8317.1984.tb00802.x

McArdle, J. J. [John J.]. (2009). Latent variable modeling of differences and changes with longitudinal data. *Annual review of psychology*, *60*, 577–605. 00289. doi:10.1146/annurev.psych.60.110707.163612

Molenaar, P. C. M. (2004, October 1). A manifesto on psychology as idiographic science: Bringing the person back into scientific psychology, this time forever. *Measurement: Interdisciplinary Research and Perspectives*, *2*(4), 201–218. doi:10.1207/s15366359mea0204_1

Mundlak, Y. (1978). On the Pooling of Time Series and Cross Section Data. *Econometrica*, *46*(1), 69–85. doi:10.2307/1913646

Neale, M. C., Hunter, M. D., Pritikin, J. N., Zahery, M., Brick, T. R., Kirkpatrick, R. M., ... Boker, S. M. (2015). OpenMx 2.0: Extended structural equation and statistical modeling. *Psychometrika*, 1–15. doi:10.1007/s11336-014-9435-8

Neale, M. C., Hunter, M. D., Pritikin, J. N., Zahery, M., Brick, T. R., Kirkpatrick, R. M., ... Boker, S. M. (2016, June). OpenMx 2.0: Extended structural equation and statistical modeling. *Psychometrika*, *81*(2), 535–549. doi:10.1007/s11336-014-9435-8. pmid: 25622929

Oravecz, Z., Tuerlinckx, F. & Vandekerckhove, J. (2009, January 10). A hierarchical Ornstein-Uhlenbeck model for continuous repeated measurement data. *Psychometrika*, *74*(3), 395–418. doi:10.1007/s11336-008-9106-8

Oravecz, Z., Tuerlinckx, F. & Vandekerckhove, J. (2016). Bayesian data analysis with the bivariate hierarchial Ornstein-Uhlenbeck process model. *Multivariate Behavioral Research*, (51), 106–119. doi:10.1080/00273171.2015.1110512

Oud, J. H. L. (2002). Continuous time modeling of the cross-lagged panel design. *Kwantitatieve Methoden*, *69*(1), 1–26. Retrieved July 30, 2014, from http://members.chello.nl/j.oud7/Oud2002.pdf

Oud, J. H. L. & Folmer, H. (2011, November 30). Reply to Steele & Ferrer: Modeling oscillation, approximately or exactly? *Multivariate Behavioral Research*, *46*(6), 985–993. doi:10.1080/00273171.2011.625306. pmid: 26736120

Oud, J. H. L. & Jansen, R. A. R. G. (2000). Continuous time state space modeling of panel data by means of SEM. *Psychometrika*, *65*(2), 199–215. doi:10.1007/BF02294374

R Core Team. (2014). *R: A language and environment for statistical computing*. Vienna, Austria: R Foundation for Statistical Computing. Retrieved from http://www.R-project.org/

Rindskopf, D. (1984, March 1). Using phantom and imaginary latent variables to parameterize constraints in linear structural models. *Psychometrika*, *49*(1), 37–47. doi:10.1007/BF02294204

RStudio Team. (2016). *RStudio: Integrated Development Environment for R*. Boston, MA: RStudio, Inc. Retrieved from http://www.rstudio.com/

Särkkä, S. (2013, September 5). *Bayesian filtering and smoothing*. Cambridge University Press.

Särkkä, S., Hartikainen, J., Mbalawata, I. S. & Haario, H. (2013, December 13). Posterior inference on parameters of stochastic differential equations via non-linear Gaussian filtering and adaptive MCMC. *Statistics and Computing*, *25*(2), 427–437. doi:10.1007/s11222-013-9441-1

Schimmack, U. (2008). The structure of subjective well-being. *The science of subjective well-being*, 97–123. 00148.

Schuurman, N. K. (2016, June 27). Multilevel autoregressive modeling in psychology: Snags and solutions. *Doctoral dissertation*. Retrieved February 24, 2017, from http://dspace.library.uu.nl/handle/1874/337475

Singer, H. (1998). Continuous Panel Models with Time Dependent Parameters. *Journal of Mathematical Sociology*, *23*(2), 77–98. doi:10.1080/0022250X.1998.9990214

Spiegelhalter, D. J., Thomas, A., Best, N. G., Gilks, W. & Lunn, D. (1996). BUGS: Bayesian inference using Gibbs sampling. *Version 0.5,(version ii) http://www. mrc-bsu. cam. ac. uk/bugs*, *19*.

Stan Development Team. (2016a). RStan: The R interface to Stan (Version 2.11). Retrieved from http://mc-stan.org

Stan Development Team. (2016b). *Stan modeling language users guide and reference manual, version 2.9.0*. Retrieved from http://mc-stan.org

Steele, J. S. & Ferrer, E. (2011, November 30). Response to Oud & Folmer: Randomness and residuals. *Multivariate Behavioral Research*, *46*(6), 994–1003. doi:10.1080/00273171.2011.625308. pmid: 26736121

Steele, J. S., Ferrer, E. & Nesselroade, J. R. (2014). An idiographic approach to estimating models of dyadic interactions with differential equations. *Psychometrika*, *79*(4), 675–700. doi:10.1007/s11336-013-9366-9. pmid: 24352513

Tokuda, T., Goodrich, B., Van Mechelen, I., Gelman, A. & Tuerlinckx, F. (2011). Visualizing distributions of covariance matrices. *Unpublished manuscript*. Retrieved March 17, 2017, from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.221.680&rep=rep1&type=pdf

Tómasson, H. (2013, November 28). Some computational aspects of Gaussian CARMA modelling. *Statistics and Computing*, *25*(2), 375–387. doi:10.1007/s11222-013-9438-9

Voelkle, M. C. & Oud, J. H. L. (2013). Continuous time modelling with individually varying time intervals for oscillating and non-oscillating processes. *British Journal of Mathematical and Statistical Psychology*, *66*(1), 103–126. doi:10.1111/j.2044-8317.2012.02043.x

Voelkle, M. C. & Oud, J. H. L. (2015, July 3). Relating latent change score and continuous time models. *Structural Equation Modeling: A Multidisciplinary Journal*, *22*(3), 366–381. doi:10.1080/10705511.2014.935918

Voelkle, M. C., Oud, J. H. L., Davidov, E. & Schmidt, P. (2012). An SEM approach to continuous time modeling of panel data: Relating authoritarianism and anomia. *Psychological Methods*, *17*(2), 176–192. doi:10.1037/a0027543. pmid: 22486576

von Oertzen, T., Brandmaier, A. M. & Tsang, S. (2015). Structural Equation Modeling With Onyx. *Structural Equation Modeling*, *22*(1), 148–161. doi:10.1080/10705511.2014.935842

Wang, Q., Molenaar, P., Harsh, S., Freeman, K., Xie, J., Gold, C., ... Ulbrecht, J. (2014). Personalized State-space Modeling of Glucose Dynamics for Type 1 Diabetes Using Continuously Monitored Glucose, Insulin Dose, and Meal Intake: An Extended Kalman Filter Approach. *Journal of Diabetes Science and Technology*, *8*(2), 331–345. doi:10.1177/1932296814524080

Wang, Z. (2013). Cts: An R Package for Continuous Time Autoregressive Models via Kalman Filter. *Journal of Statistical Software*, *53*(5), 1–19.

Xie, Y. (2014). Knitr: A Comprehensive Tool for Reproducible Research in R. In V. Stodden, F. Leisch & R. D. Peng (Eds.), *Implementing Reproducible Computational Research*. Chapman and Hall/CRC. Retrieved from http://www.crcpress.com/product/isbn/9781466561595

Appendix A.

Package comparisons

The following script loops over a sequence in which data is generated from a very simple model (to facilitate comparison), then various packages are used to fit the data, and the distribution of parameter estimates is then plotted. ctsem is used to generate 500 time points of data for a single individual, for a single perfectly measured, stationary process with a drift value of -0.3. The different packages at times use different transformations of the parameters, but for ease of comparison these are transformed to the drift parameter in ctsem. This comparison is not intended as a critique of any package, differences in intended use and estimation routines may result in the observed differences, and it is hoped the provided code may help others to explore alternatives when necessary.

```
  if (!requireNamespace("cts", quietly = TRUE)) {
    stop("cts package needed for this function to work. Please install it.",
      call. = FALSE)
  }
  if (!requireNamespace("yuima", quietly = TRUE)) {
    stop("yuima package needed for this function to work. Please install it.",
      call. = FALSE)
  }
Tpoints<-500
output <- matrix(NA, 100, 6)
colnames(output) <- c("True", "ctsem", "cts", "yuima", "arima", "OpenMx")
for(i in 1:nrow(output)){
generatingModel <- ctModel(n.latent = 1, n.manifest = 1,
  Tpoints = Tpoints,
  LAMBDA = diag(1), DRIFT = matrix(-.3, nrow = 1),
  CINT = matrix(3, 1, 1),
  MANIFESTVAR = diag(0.00001, 1),
  DIFFUSION = t(chol(diag(5, 1))))

output[i, 1] <- generatingModel$DRIFT #true value

ctsemData <- ctGenerate(generatingModel, n.subjects=1, burnin=30)
longData <- ctWideToLong(ctsemData, Tpoints=500, n.manifest=1)
longData <- ctDeintervalise(longData)

### ctsem package
ctsemModel <- ctModel(n.latent=1, n.manifest = 1,
  Tpoints = Tpoints,
  MANIFESTVAR = diag(0.0001, 1),
  LAMBDA = diag(1))
ctsemFit <- ctFit(ctsemData, ctsemModel, stationary = c("T0VAR"))
output[i,2] <- mxEval(DRIFT, ctsemFit$mxobj)

### CTS package
ctsData <- longData[, c("time", "Y1")]
library(cts)
ctsFit <- car(ctsData, order = 1, scale = 1)
output[i, 3] <- -1 * (1 + ctsFit$phi) / (1 - ctsFit$phi)
```

```
### yuima package (not plotted - potential issues due to dT=1)
library(yuima)
mod <- setModel(drift="drift * x + cint", diffusion = "diffusion")
ou <- setYuima(model = mod, data = setData(longData[,"Y1"], delta = 1))
mlout <- qmle(ou,start = list(drift = -.3, diffusion = 1, cint = 1))
output[i, 4] <- mlout@coef[2]

### arima (from stats package, discrete time analysis only)
arfit <- arima(longData[, "Y1"], order = c(1, 0, 0))
log(arfit$coef[1]) #transform ar1 parameter to drift parameter
output[i, 5]<-log(arfit$coef[1])

### OpenMx state space function
ctsemModel <- ctModel(n.latent=1, n.manifest = 1,
  Tpoints = Tpoints,
  MANIFESTVAR = diag(0.0001, 1), TOVAR=diag(1), LAMBDA = diag(1))
cdim <- list("Y1", c("eta1"))

amat <- mxMatrix("Full", 1, 1, TRUE, .3,
name="A", lbound=-10)
bmat <- mxMatrix("Full", 1, 1, FALSE, 0, name="B")
cmat <- mxMatrix("Full", 1, 1, FALSE, 1, name="C", dimnames=cdim)
dmat <- mxMatrix("Full", 1, 1, TRUE, 1, name="D")
qmat <- mxMatrix("Full", 1, 1, TRUE, 1, name="Q")
rmat <- mxMatrix("Diag", 1, 1, FALSE, .0001, name="R")
xmat <- mxMatrix("Full", 1, 1, TRUE, 7, name="x0", lbound=-10, ubound=10)
pmat <- mxMatrix("Diag", 1, 1, FALSE, 1, name="P0")
umat <- mxMatrix("Full", 1, 1, FALSE, 1, name="u")
tmat <- mxMatrix("Full", 1, 1, name="time", labels="data.time")

osc <- mxModel("AR1",
amat, bmat, cmat, dmat, qmat, rmat, xmat, pmat, umat, tmat,
mxExpectationStateSpace("A", "B", "C", "D", "Q", "R", "x0", "P0", "u"),
mxFitFunctionML(),
mxData(longData, "raw"))

oscr <- mxRun(osc)

output[i,6] <-  log(mxEval(A, oscr))

} #end for loop

### plot output
plot(density(output[1:i, 2]), xlim = c(-.4, -.1), lty = 2, lwd = 2,
  ylab="Density",
  main = "Density of estimates of drift parameter (true value -0.3)")
points(density(output[1:i, 3]), col="red", type="l", lty=3, lwd=3)
points(density(output[1:i, 5]), col="blue", type="l", lwd=2, lty=2)
points(density(output[1:i, 6]), col="green", type="l", lwd=4, lty=5)
legend("topleft", bty="n",
  text.col = c("black","red","blue", "green"),
  legend = c("ctsem", "cts", "arima", "OpenMx"))
```

Appendix B.

Dynamic model of wellbeing — R script

This script installs and loads ctsem, and specifies the dynamic model of wellbeing used in this paper (although we cannot distribute the GSOEP data). The script also plots the priors used for the population level model, as well as examples of possible priors for the subject level parameters – because the subject level priors depend on the estimated population level parameters.

```
install.packages("ctsem")
require(ctsem)

model<-ctModel(type="stanct", LAMBDA=diag(2),
  n.manifest=2, manifestNames=c("overallSat", "healthSat"),
  n.latent=2, latentNames=c("overallSat", "healthSat"),
  n.TIpred=2, TIpredNames=c("meanAge", "meanAgeSq"))

plot(model, wait=TRUE)

fakedata=matrix(1,nrow=5,ncol=6)
colnames(fakedata)=c("id","time","overallSat","healthSat","meanAge","meanAgeSq")
fakedata[,"time"]=1:5
fit<-ctStanFit(fakedata,model,fit=FALSE)
cat(fit$stanmodeltext)
```

Appendix C.

Time independent predictor effects

Table C.1.: Posterior distributions for time independent predictor effects

	2.5%	mean	97.5%
Age_T0mean_overallSat	-0.31	-0.05	0.21
Age_T0mean_healthSat	-0.17	0.12	0.40
Age_drift_overallSat_overallSat	-0.21	-0.06	0.10
Age_drift_overallSat_healthSat	-0.08	-0.01	0.07
Age_drift_healthSat_overallSat	-0.06	-0.01	0.03
Age_drift_healthSat_healthSat	-0.15	-0.03	0.09
Age_diffusion_overallSat_overallSat	-0.18	-0.05	0.08
Age_diffusion_healthSat_overallSat	-0.02	0.07	0.17
Age_diffusion_healthSat_healthSat	-0.04	0.09	0.20
Age_manifestvar_overallSat_overallSat	-0.04	0.03	0.09
Age_manifestvar_healthSat_healthSat	-0.04	0.02	0.09
Age_manifestmeans_overallSat	-0.11	0.07	0.25
Age_manifestmeans_healthSat	-0.63	-0.39	-0.15
Age_T0var_overallSat_overallSat	-0.33	0.19	1.01
Age_T0var_healthSat_overallSat	-0.49	0.16	0.79
Age_T0var_healthSat_healthSat	-0.23	0.05	0.45
AgeSq_T0mean_overallSat	-0.14	0.11	0.36
AgeSq_T0mean_healthSat	-0.26	0.04	0.33
AgeSq_drift_overallSat_overallSat	-0.25	-0.11	0.02
AgeSq_drift_overallSat_healthSat	-0.10	-0.04	0.02
AgeSq_drift_healthSat_overallSat	-0.01	0.02	0.06
AgeSq_drift_healthSat_healthSat	-0.08	0.01	0.10
AgeSq_diffusion_overallSat_overallSat	-0.10	0.02	0.14
AgeSq_diffusion_healthSat_overallSat	-0.05	0.02	0.12
AgeSq_diffusion_healthSat_healthSat	-0.17	-0.07	0.04
AgeSq_manifestvar_overallSat_overallSat	-0.07	0.00	0.06
AgeSq_manifestvar_healthSat_healthSat	-0.06	0.00	0.07
AgeSq_manifestmeans_overallSat	-0.20	-0.01	0.18
AgeSq_manifestmeans_healthSat	-0.15	0.10	0.33
AgeSq_T0var_overallSat_overallSat	-1.26	-0.32	0.18
AgeSq_T0var_healthSat_overallSat	-0.69	-0.00	0.65
AgeSq_T0var_healthSat_healthSat	-0.19	0.10	0.38

Appendix D.

Plotting discrete time effects from continuous time parameters

```
require(ctsem)
nlatent=2 #number of latent processes
dt=seq(.1,15,.1) #vector of time intervals

#Basic continuous time parameters
DRIFT=matrix(c(-.4,0,.2,-.2),nrow=nlatent,ncol=nlatent)
CINT<-matrix(c(3,-4),ncol=1)
DIFFUSION=matrix(c(2,-1,-1,3),nrow=nlatent,ncol=nlatent) # in covariance form (Q matrix)

#implied discrete DRIFT (auto and cross regression matrix) given time intervals
discreteDRIFT=array(unlist(lapply(dt,function(x) {
  expm(DRIFT * x)
  })),dim=c(nlatent,nlatent,length(dt)))

plot(dt, discreteDRIFT[1,1,],type="l",lwd=2,col="blue",
  ylim=c(min(discreteDRIFT),max(discreteDRIFT)))
points(dt, discreteDRIFT[2,2,],type="l",lwd=2,col="red")
points(dt, discreteDRIFT[2,1,],type="l",lwd=2,col="green")
points(dt, discreteDRIFT[1,2,],type="l",lwd=2,col="purple")

#implied discrete time intercept given time intervals
discreteCINT <- array(unlist(lapply(dt,function(x) {
  solve(DRIFT)%*%(expm(DRIFT * x) - diag(2)) %*% CINT
  })),dim=c(nlatent,length(dt)))

matplot(dt,t(discreteCINT),type="l",lwd=2)

#asymptotic level of the processes (if stationary)
asymCINT <- -solve(DRIFT) %*% CINT

#asymptotic within subject process covariance matrix
asymDIFFUSION<-matrix(-solve(
  DRIFT %x% diag(nrow(DRIFT)) + diag(nrow(DRIFT)) %x% DRIFT) %*%
    c(DIFFUSION), nrow=nlatent,ncol=nlatent)

#implied discrete time DIFFUSION (latent process covariance) matrix given time intervals
discreteDIFFUSION=array(unlist(lapply(dt,function(x) {
  matrix(asymDIFFUSION - (expm(DRIFT * x) %*%
      asymDIFFUSION %*%
      t(expm(DRIFT * x))),nrow=nlatent,ncol=nlatent)
})),dim=c(nlatent,nlatent,length(dt)))

plot(dt, discreteDIFFUSION[1,1,],type="l",lwd=2,col="blue",
  ylim=c(min(discreteDIFFUSION),max(discreteDIFFUSION)))
points(dt, discreteDIFFUSION[2,2,],type="l",lwd=2,col="red")
points(dt, discreteDIFFUSION[1,2,],type="l",lwd=2,col="purple")
```

Appendix E.

Simulation of hierarchical correlation matrices

```
require(rstan)
sm <- "
functions{

row_vector vecpower(row_vector vec, real power){
row_vector[num_elements(vec)] out;
for(i in 1:num_elements(vec)){
out[i] = vec[i]^power;
}
return out;
}

row_vector vecsqrt(row_vector vec){
row_vector[num_elements(vec)] out;
for(i in 1:num_elements(vec)){
out[i]=sqrt(vec[i]);
}
return out;
}

matrix cholcor(matrix mat){ //converts from lower partial cor matrix to cholesky cor
int ndim;
matrix[rows(mat),cols(mat)] mcholcor;
ndim = cols(mat);
mcholcor=diag_matrix(rep_vector(0,ndim));
mcholcor[1,1]=1;

if(ndim > 1){
for(coli in 1:ndim){
for(rowi in coli:ndim){
if(coli==1 && rowi > 1) mcholcor[rowi,coli] =  mat[rowi,coli];
if(coli > 1){
if(rowi == coli) mcholcor[rowi,coli] = prod(vecsqrt(1-vecpower(mat[rowi,1:(coli-1)],2)));
if(rowi > coli) mcholcor[rowi,coli] = mat[rowi,coli] *
prod(vecsqrt(1-vecpower(mat[rowi,1:(coli-1)],2)));
}
}
}
}
return mcholcor;
}

} //end functions

data{
int ndim; //dimension of correlation matrices
int N; //number of subjects
real pcorrvechsdscale; //sd of raw pop means
}
```

```
parameters{
vector[(ndim*ndim-ndim)/2] pcorrvec[N]; //subjects partial correlation parameter vectors
vector[(ndim*ndim-ndim)/2] pcorrvecmu; //pop mean of partial correlation vectors
vector<lower=0>[(ndim*ndim-ndim)/2] pcorrvechsd; //pop sd of partial correlation vectors
}

transformed parameters{
matrix[ndim,ndim] pcorrmat[N]; //subjects partial correlation matrices
matrix[ndim,ndim] cholcorrmat[N];

{
int counter;
for(n in 1:N){
counter=0;
pcorrmat[n]=diag_matrix(rep_vector(0,ndim));
for(coli in 1:(ndim-1)){ // fill lower tri of pcorrmat
for(rowi in (coli+1):ndim){
counter=counter+1;
pcorrmat[n][rowi,coli]=inv_logit(pcorrvecmu[counter]*1.5 +
pcorrvechsd[counter] * pcorrvec[n][counter])*2-1;
}
}
cholcorrmat[n] = cholcor(pcorrmat[n]);
}
}

}
model{
pcorrvecmu~normal(0,1);
pcorrvechsd~normal(0,pcorrvechsdscale);
for(n in 1:N){
pcorrvec[n]~normal(0,1);
}

//adjust pop mean partial correlation probabilities
{
real betad;
int counter;

betad = 1.5 + (ndim-1)/2;

if(ndim > 1){
counter=0;
for(coli in 1:(ndim-1)){
betad = betad - .6;
for(rowi in (coli+1):ndim){
counter=counter+1;
inv_logit(pcorrvecmu[counter]*1.5) ~ beta(betad,betad);
}
}
}
}
}
generated quantities{
matrix[ndim,ndim] corrmat_out[N];
for(n in 1:N){
corrmat_out[n] = cholcorrmat[n] * cholcorrmat[n]\'; //the full correlation matrix
}
}
"

N=10
```

```
ndim=6
eta=1.5


fit=stan(model_code=sm,init=0,data=list(ndim=ndim,eta=eta,N=N,
pcorrvechsdscale=.3),warmup=1000,iter=30000,chains=3,cores=3)

e=extract(fit)

colourvec<-rainbow(n=ndim) #colours indicate the column of the correlation matrices

#plot the prior densities for correlations on top of each other
for(n in 1:N){
counter=0
for(coli in 1:(ndim-1)){
for(rowi in (coli+1):ndim){
counter=counter+1
if(n ==1 & coli ==1 & rowi==2) {
plot(density(e$corrmat_out[,n,rowi,coli],bw=.1),col=coli,xlim=c(-1.1,1.1))
} else {
points(density(e$corrmat_out[,n,rowi,coli],bw=.1),col=coli,type="l")
}
}}}
```